Reliable Transaction Router

System Manager's Manual

Order Number: AA-Q88CE-TE

June, 1999

This manual describes how to configure, manage and monitor Reliable Transaction Router, Version 3.2 (RTR).

Revision/Update Information: This manual supersedes Version 3.1D of the *System Manager's Manual*

Software Version: Reliable Transaction Router, Version 3.2

Compaq Computer Corporation
Houston, Texas

June, 1999

This document was prepared using VAX DOCUMENT, Version 2.1.

Contents

4 Transaction Management

5 RTR Monitoring

6 RTR Command Line Interface

Figures

Tables

Preface

Purpose of this Manual

This manual describes how to configure, manage and monitor the operation of Reliable Transaction Router (RTR) using the RTR Command Line Interface (CLI).

Intended Audience

The *System Manager's Manual* is intended for persons who perform system management functions to configure, test, monitor and maintain RTR applications.

The reader is assumed to be familiar with their operating system, but not necessarily experienced with RTR operations.

New users of RTR are encouraged to read the first chapters of the *Application Programmer's Reference Manual* for a overall description of RTR.

Structure of Document

The manual contains the following chapters and appendices:

- Chapter 1 is an introduction to the RTR Command Line Interface (CLI) and tells you how to use local and remote commands and command procedures.
- Chapter 2 explains how to start and configure RTR.
- Chapter 3 describes RTR's Partition Management.
- Chapter 4 gives an overview of RTR's Transaction Management tools.
- Chapter 5 describes how the RTR monitor is used to continuously observe the performance and operation of RTR and applications using RTR.
- Chapter 6 describes how to use the CLI interface to RTR API calls and contains a complete description of all RTR commands, listed alphabetically.
- Appendix A tells you how to create your own monitor pictures for special monitoring needs.
- Appendix B describes server shadowing and recovery.
- Appendix C describes how to use RTR with XA and ORACLE.
- Appendix D contains a list of all the error messages that can be returned by the RTR CLI.
- Appendix E contains a list of all messages that RTR can write to the RTR log.

Related Documentation

- *Release Notes*
- *Installation Guide*
- *Application Programmer's Reference Manual*
- *Application Design Guide*
- *Migration Guide*

Reader's Comments

Compaq welcomes your comments on this manual. Please send us your comments by email to rtrdoc@compaq.com.

Include the title of the manual, section and page numbers with your comments or suggestions.

Conventions

Table 1 describes the conventions used in this guide.

Table 1 Conventions Used in this Guide

Convention	Meaning
UPPERCASE lowercase	Some operating systems differentiate between lowercase and uppercase characters. For these systems, examples, syntax descriptions, function definitions, and literal strings that appear in text must be typed exactly as shown. Commands typed to the RTR CLI are *not* case sensitive unless enclosed in quote marks
#	A number sign (#) is the default operating system superuser prompt.
%	A percent sign (%) is the default operating system user prompt on UNIX systems.
$	A dollar sign ($) is the default operating system user prompt on OpenVMS systems.
Return	In examples, a boxed symbol indicates that you must press the named key on the keyboard.
Ctrl/C	This symbol indicates that you must press the Ctrl key while you simultaneously press another key (in this case, C).
user input	In interactive examples, this typeface indicates input entered by the user.
filesystem	In text, this typeface indicates the exact name of a command, routine, partition, pathname, directory, or file. This typeface is also used in interactive examples and other screen displays.
italic text	Italic text emphasizes important information, indicates variables, and indicates complete titles of manuals. Italic text also represents information that can vary in system messages (for example, Internal error *number*), command lines (for example, /PRODUCER=*name*), and command parameters in text.
boldface text	Boldface text represents the introduction of a new term or the name of a command, an argument, an attribute, or a reason.
[y]	In a prompt, square brackets indicate that the enclosed item is the default response. For example, [y] means the default response is Yes.
text	A vertical bar next to the text | indicates changes or additions since the previous version of this document.
HTML	Red HTML text indicates changes or additions since the previous version of this document.

1
Introduction

For a general introduction to Reliable Transaction Router, Version 3.2 (RTR), you should read the introductory chapter in the Reliable Transaction Router *Application Design Guide*. Additional information about the Reliable Transaction Router is available in the Reliable Transaction Router *Application Programmer's Reference Manual*.

In order to use RTR, you must install the RTR software and your application. See the Reliable Transaction Router *Installation Guide* for instructions for installing RTR.

1.1 Getting Started

RTR applications use the API calls described in the Reliable Transaction Router *Application Programmer's Reference Manual*. Before an RTR application can be used, RTR must be started on every node in your RTR network. You do this is by issuing a START RTR command on each node. You may wish to include the START RTR command in a startup command procedure for each node, so that RTR is started whenever a node is booted.

Many applications can use RTR at the same time without interfering with one another. This is achieved by defining a separate **facility** for each application. A facility can be thought of as an application's own runtime environment of RTR. (In addition, distributed RTR applications may start and execute many transactions. RTR is capable of massively parallel operation.)

Before application processes are started, a facility must be defined using the CREATE FACILITY command. You may wish to include the CREATE FACILITY command to the command procedure used to start the application.

The rest of this chapter explains how to use RTR commands. The START RTR and CREATE FACILITY commands are described in detail in Chapter 2 and Chapter 6.

1.2 Entering Commands

RTR is started, configured and maintained by using the **RTR Command Line Interface** (CLI). The RTR CLI is used to start, set up and monitor the operation of RTR.

The RTR CLI is accessed by entering RTR at the operating system prompt. Commands can either be entered on the same line as the RTR verb, for example:

```
% rtr command
```

or, when several commands are to be entered at the RTR prompt:

```
% rtr
RTR> start rtr
RTR> create journal
```

Note

For convenience, the user prompt for the operating system is shown here as the "%" symbol. Your system may have a different prompt.

The RTR CLI accepts commands that you type and can process procedures consisting of RTR commands.

Most RTR commands accept qualifiers: these are indicated by the forward slash (/) character. For example, many RTR commands accept the "/OUTPUT" qualifier; it directs the output from the command to a file.

The forward slash (/) character may also appear in the filenames of some operating systems; such filenames must be enclosed in quotation marks to ensure that RTR does not interpret the filename as a command qualifier.

When RTR commands are entered on a single line, you may need to use extra quotation characters, depending on the operating system in use. For example, when running on most UNIX platforms, additional single quotation marks are required when entering quoted items such as filenames. Compare the following commands.

```
% rtr
RTR> show facility/output="/usr/users/test/fac_output.lis"
```

```
% rtr show facility/output='"/usr/users/test/fac_output.lis"'
```

The first command works for OpenVMS systems but not on UNIX. The second command uses single quotation marks outside of double quotations to be correctly interpreted on UNIX systems.

1.3 Online Help

You can get information about the RTR CLI by using the HELP command. Entering the following command:

```
% rtr help
```

displays a complete list of help topics on your terminal.

If you require additional information then enter the topic directly on the same line, for example:

```
% rtr help show
```

The help command can also be used to find out about errors returned by RTR. The folowing sequence returns the error identifier:

```
% rtr help errors error-identification
```

where error-identification is the identification part of the returned error. The following sequence returns an error message, RTRALRSTA, that can then be explained by the help errors rtralrsta command option:

```
%  rtr
RTR> start rtr
%RTR-F-RTRALRSTA, rtr already started
RTR> help errors rtralrsta

Errors

  RTRALRSTA

RTR already started

    Explanation: RTR was already running when the "START RTR" command
    was executed. This error message is displayed by the RTR utility.

RTR>
```

1.4 Command Procedures

RTR commands can also be written in a command file and then executed as a procedure using the EXECUTE file-spec or @file-spec commands. For example:

```
% rtr execute createfacil
```

or:

```
% rtr @createfacil
```

or at the RTR prompt:

```
% rtr
RTR> execute createfacil
```

or:

```
RTR> @createfacil
```

1.5 Remote Commands

Most RTR commands can be issued either locally (the default) or on one or more remote nodes. To be able to issue commands to a remote node you must have an account on that node with the necessary access privileges. Refer to your operating system documentation for information on how to set up the access privileges.

To specify the remote node names explicitly:

```
RTR> command/node=node-list
```

To specify remote nodes implicitly, if for example the command is to be executed in every node of a clustered environment use a command of the following form:

```
 RTR> command/cluster
```

Examples:

```
RTR> start rtr/node=(nodeA,nodeB,nodeC)
```

This command starts RTR on the three nodes.

Note

The /CLUSTER and /NOCLUSTER command qualifiers refer to cluster support. These qualifiers are for operating systems that fully support clustering. Use of the /CLUSTER qualifier on systems that do not have clustering causes the relevant command to be executed on the local node only. For example Windows 95 systems do not support clustering.

If several commands need to be executed remotely on the same nodes then the `set environment` command can be used to save typing.

For example:

```
% RTR
RTR> set environment/node=(nodeA, nodeC)
RTR> stop rtr
```

This example shows the use of the `set environment` command to stop rtr on Node A and Node C. More details concerning the commands used in the above examples are contained in Section 6.2.

2
Starting and Setting Up RTR

This chapter describes how to configure and start an RTR environment. Recovery journals, router load balancing and call-out servers are also discussed.

2.1 Introduction

Before RTR applications can run, RTR must be started and the application's `facility` must be defined on each node of the application's environment. This is done by issuing the `start rtr` and `create facility` commands on each participating node. There are several ways to accomplish this:

- You can log on to each node in turn and issue the commands interactively.
- You can log on to one node and use the remote command capability to configure all the nodes from one session.
- You can include the necessary commands in a startup script or command file on each node so the commands are automatically executed when the nodes are booted.

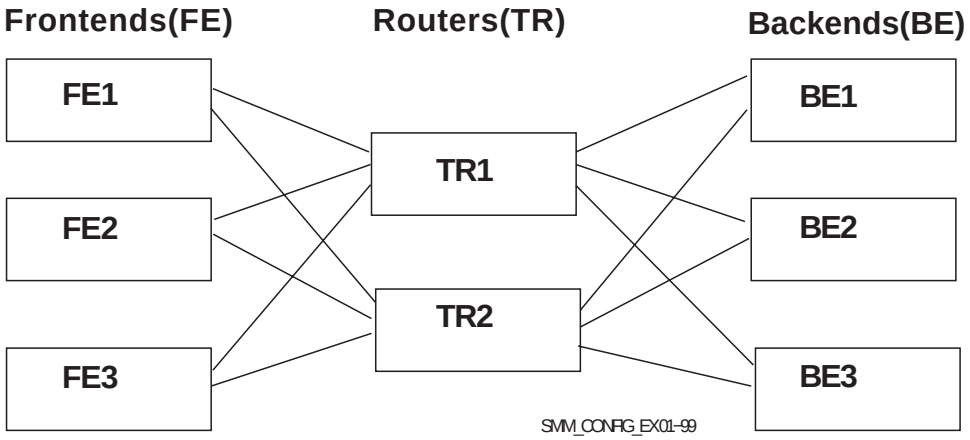
The first two methods are more suited to a development or test environment, the last method more suited to a production environment.

The remaining sections contain examples of the commands that are used to start and configure RTR. Section 6.2 gives syntax details of the RTR commands.

2.2 Setting Up—An Example

The following example assumes that RTR is started on the eight node system shown in Figure 2–1.

Figure 2–1 Configuration Example



In this example, the application client processes run on the nodes FE1, FE2 and FE3. The servers run on BE1, BE2 and BE3. Nodes TR1 and TR2 are routers and have no application processes running on them. This diagram shows all possible connections. The frontend connects to only one router at a time.

Example 2–1 shows the commands that have to be issued on each node to start this configuration. Commands are issued first on node FE1, then on FE2, and on FE3 for frontends followed by TR1 and TR2 for routers, and finally BE1, BE2, and BE3 for backends.

Example 2–1 Local Configuration of each Node

```
% rtr
RTR> start rtr

RTR> create facility funds_transfer/frontend=FE1 -
_RTR>                            /router=(TR1, TR2)

% rtr
RTR> start rtr

RTR> create facility funds_transfer/frontend=(FE1, FE2, FE3) -
_RTR                             /router=TR1 -
_RTR>      /backend=(BE1, BE2, BE3)

% rtr
RTR> start rtr

RTR> create facility funds_transfer/router=(TR1, TR2) -
_RTR>      /backend=BE1
```

The commands shown in Example 2–1 could also be included in each node's startup script or put in a command procedure used to start the application.

Note that nodes only need to know about the nodes in the neighbouring layers of the hierarchy, thus FE1 does not need to know about BE1. Superfluous node names are ignored. This allows you to issue the same CREATE FACILITY command on every node to simplify the maintenance of startup command procedures.

Example 2–2 illustrates how to use RTR remote commands to start the same configuration. The set environment command is used to send subsequent commands to a number of RTR nodes.

Example 2–2 Remote Setup from one Node

```
% rtr
RTR> set environment/node= -
_RTR>    (FE1, FE2, FE3, TR1, TR2, BE1, BR2, BR3)

RTR> start rtr

RTR> create facility funds_transfer/frontend=(FE1, FE2, FE3) -
_RTR>                           /router=(TR1, TR2) -
_RTR>                           /backend=(BE1, BE2, BE3)
```

You can enter the commands shown in Example 2–2 on any of the nodes in the configuration. However, you must have an account with the necessary privileges on the other nodes.

To find out if RTR has been started on a particular node, use the SHOW RTR command.

To find out which facilities have been created (if any) and how they are configured you can use the SHOW FACILITY and SHOW LINK commands. The full syntax of these commands is given in Chapter 6.

2.3 Creating a Recovery Journal

RTR writes data to journal files to be able to recover (that is, replay) partially executed transactions after a backend node failure.

For performance reasons, the journal may be spread across several disks. Specify the location and size of the journal using the CREATE JOURNAL command.

The CREATE JOURNAL command must be issued on each node where an application server will run. That is, on each backend node and on any router nodes where router call-out servers will run. It must be issued after installing RTR and before creating any facilities.

It may be issued again later to alter the size or location of the journal to improve performance. Use the MODIFY JOURNAL command to adjust journal sizes.

Cautionary Note for Journals

- The CREATE JOURNAL/SUPERSEDE command deletes the contents of any existing journal files. If transaction recovery is required, DO NOT ISSUE this command after a failure.
- Do not make backup copies of journal files without first making the original journal file read-only or the journal files will be considered spurious by RTR because it sees journal files that it did not create. In this case RTR will issue a %RTR-F-SPUJOUFIL error message.

- The operator should move any duplicate copies of journal files to a location other than the `rtrjnl/groupname` directory so that RTR will see only the one it created.
- Track duplicate copies of journal files in the log file to prevent RTR seeing more than the one it created and issuing the SPUJOUFIL error message.
- If it is determined that a journal is SPURIOUS by means other than improper copying, then a backup copy followed by a destruction of the transactions contained in a journal can be performed by the `CREATE JOURNAL/SUPERCEDE` command.

2.4 Changing a Facility

RTR facilities can be altered either by deleting and re-creating them using the `delete facility` `create facility` commands. Alternatively, you can use the `extend facility` and `trim facility` commands.

With sufficient redundancy in a system, facility modification can be carried out with minimal interruption to the application.

Note

The RTR facility defines the nodes used by an application, and the roles (frontend, router, backend) they may assume. You do not need to change facility definitions in the event of node or link failures.

In the example in Figure 2–1, assume that the FE3 node is being removed from the FUNDS_TRANSFER facility. Since FE3 is a frontend for this facility, only the routers (TR1 and TR2) need be reconfigured. The routers can be reconfigured one after another, so that the service provided to the remaining frontends (FE1 and FE2) is not interrupted.

Example 2–3 shows the `delete facility` and `create facility` commands that are issued from node BE1 (for example), in order to achieve this reconfiguration.

Example 2–3 Reconfiguration Using Delete and Create Facility

```
% rtr
RTR> stop rtr/node=FE3 1
RTR> delete facility funds_transfer/node=TR2 2
RTR> create facility funds_transfer/node=TR2 - 3
_RTR>                          /frontend=(FE1,FE2) -
_RTR>                          /router=TR2 )
RTR> delete facility funds_transfer/node=TR1  4
RTR> create facility funds_transfer/node=TR1 - 5
_RTR>                          /frontend=(FE1,FE2) -
_RTR>                          /router=TR1 )
```

1 RTR is stopped on node FE3, the node being excluded from the network. In order to prevent transactions being interrupted or aborted, application processes should be stopped in an orderly manner before issuing the 'stop rtr' command. Alternatively, a `stop rtr /abort` command will force application processes using RTR to exit, aborting or interrupting any outstanding transactions.

2 The facility is deleted on node TR2. Any frontends that were connected to TR2 will connect to the remaining router, node TR1.

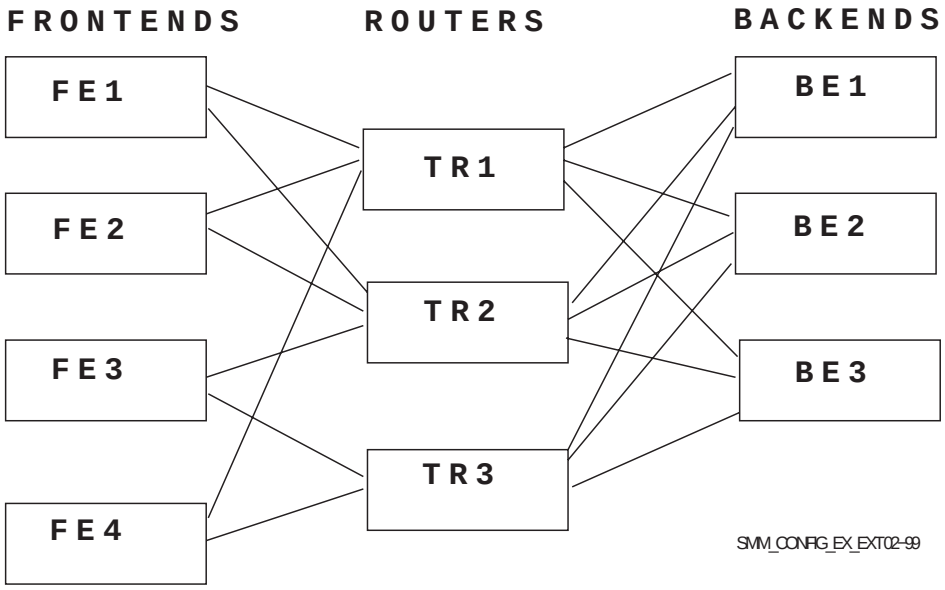
3 The facility is created on node TR2, excluding node FE3 from the frontend list. This facility has started when the start message appears in the RTR log.

4 The facility is deleted from node TR1. Frontends FE1 and FE2 now connect to router TR2.

5 The new facility is created on node TR1, again excluding node FE3 from the frontend list. The frontends now distribute their connections to the router nodes TR1 and TR2 according to the load sharing algorithm. The system is again fully operational.

In the example in Figure 2–1, assume that a new router node TR3 and new frontend FE4 are being added to the facility funds_transfer. The extended configuration is shown in Figure 2–2. This diagram shows all possible connections. The frontend connects to only one router at a time.

Figure 2–2 Extend Configuration Example



All backend nodes must be informed when router configurations are changed. Because TR3 will be a router for the FE3 and FE4 frontends, these nodes must also be informed of its presence. Likewise, TR3 must be informed about FE3 and FE4.

Example 2–4 shows the `extend facility` command used for this reconfiguration.

Example 2–4 Reconfiguration Using Extend Facility

```
% RTR
RTR> start rtr /node=(TR3,FE4)
RTR> set environment/node= -  1
_RTR> (FE1,FE2,FE3,TR1,TR2,BE1,BE2,BE3,TR3,FE4)

RTR> extend facility funds_transfer -    2
_RTR> /router=TR3/frontend=(FE3,FE4) -
_RTR> /backend=(BE1,BE2,BE3)

RTR> extend facility funds_transfer -    3
_RTR> /router=TR1/frontend=FE4
```

1 The set environment is used to send the following command to all nodes in the facility, including the new nodes.

2 The extend facility defines the new router TR3 and the new frontend FE4. Because the new router is also connected to the existing frontend FE3, this node must also be specified as a frontend. The new router TR3 is told about all backends with the /backend qualifier. Note that the extend facility command has been used to create new definitions on TR3 and FE4, and extend the definitions on BE1, BE2 and BE3.

3 The extend facility command is used to extend the definitions on TR1 and FE4 in order to give FE4 a second router link.

2.5 Setting up Callout Servers

Callout servers are applications that receive a copy of every transaction passing through the node where the callout server is running.

Like any other server, callout servers have the ability to abort any transaction that they participate in. Callout servers are typically used to provide an additional security service in the network; transactions can be inspected by the callout server and aborted if they fail to meet any user-defined criteria. Callout servers can run on router or backend nodes, or both.

Assume that callout servers are to run on the router nodes (TR1 and TR2) in the example configuration shown in Figure 2–1. Example 2–5 shows the commands needed to set up callout servers on the routers.

Example 2–5 Configuration of Callout Servers

```
% rtr
RTR> set environment/node= -
_RTR>    (FE1,FE2,FE3,TR1,TR2,BE1,BE2,BE3)

RTR> start rtr

RTR> create facility funds_transfer/frontend=(FE1,FE2,FE3) -
_RTR>                            /router=(TR1,TR2) -
_RTR>                            /backend=(BE1,BE2,BE3) -
_RTR>                            /call_out=router
```

2.6 Router Load Balancing

Router load balancing, or intelligent re-connection of frontends to a router is possible, allowing a frontend to select the router that has least loading. The `create facility` and `set facility` commands have the `/balance` qualifier to control this. RTR now allows frontends to determine their router connection. The RTR version 2 implementation of load balancing treated all routers as equal and this could cause reconnection timeouts in geographically distant routers.

When used with `create facility`, `/balance` specifies that load balancing is enabled for frontend/router connections across the facility.

Use the `set facility/[no]balance` to switch load-balancing off and on.

The default behavior (/nobalance) is for a frontend to connect to the preferred router. Preferred routers are selected in the order specified in the `create facility/router=tr1,tr2,tr3..` qualifier. Automatic failback ensures that the frontend will reconnect to the first router in the order when it becomes available. Manual balancing can be attained by specifying different router orders across the frontends.

Non load-balanced frontend connections will fail back to the preferred router when it becomes available.

Automatic load balancing institutes a router list with a random value for the frontend assigned at the time the `create facility` with the `/balance` command is issued. The frontend will sort the list of routers based on its own random order. Randomness ensures that there will be a balance of load in a configuration with a large number of frontends. Automatic failback will maintain the load distribution on the routers and failback is controlled at a limited rate so as not to overload configurations with a small number of routers.

The following points should be born in mind when using load-balancing:

- Frontends connect to a single router, per facility
- When several routers are configured on the frontend:
  - The specified list in the `create facility` constitutes the preferred search order, by default, unless `/balanced` is used
- Balancing may be individually enabled or disabled on the frontend nodes.
- Balancing is dynamic. The loss or addition of a router node causes the frontend nodes to redistribute their connections.

- Use /balance on frontend nodes only. Use of /balance on routers is supported only to enable RTR Version 2 balancing. Use this qualifier only when you are connecting frontend nodes running RTR Version 2. See CREATE FACILITY and SET FACILITY for more information on /balance.
- Commands to set/show load balancing are:
  - create facility /balance
    - Enables load balancing attribute on the executor node
    - Significant only on router and frontend nodes
    - Disabled, by default
  - set facility /[no]balance
    - Toggles the attribute setting on the executor node
  - show facility /configuration
    - Shows the current setting of the load balance attribute
  - show facility /link
    - On a frontend, shows the current router node
    - On a router, shows the frontends connected, and the current load coordinating backend node
  - show facility /balance
    - On a router node, shows the current number of frontends connected, and the current credit
    - On the coordinating backend node, shows the total number of routers, frontends and credit given out
    - Useful for troubleshooting frontend connection problems

2.7 RTR Privileges

RTR supports two levels of rights or privileges, rtroper and rtrinfo (on UNIX® platforms), RTR$OPERATOR and RTR$INFO (on OpenVMS) and RtrOperator and RtrInfo on Windows NT. In general, rtroper or RTR$OPERATOR is required to issue any command that affect the running of the system, and rtrinfo or RTR$INFO is required for using monitor and display commands.

Setting RTR Privileges on UNIX Systems

On UNIX machines RTR privileges are determined by the user id and group membership. For RTR users and operators, create the group rtroper and add RTR operators and users as appropriate.

The root user has all privileges need to run RTR. Users in the group rtroper also have all privileges with respect to RTR, but may not have sufficient privilege to access resources used by RTR, such as shared memory or access to RTR files.

The rtrinfo group is currently only used to allow applications to call rtr_request_info( ).

Depending on your UNIX system, see the addgroup, groupadd or mkgroup commands or the System Administration documentation for details on how to add new groups to your system.

The rtrinfo group is currently only used to allow applications to call rtr_request_info( ) For other users, create the groups rtroper and rtrinfo Users who do not fall into the above categories, but are members of the rtrinfo group can only use RTR commands that display information (SHOW, MONITOR, call rtr_request_info, etc.).

If the groups rtroper and rtrinfo are not defined, then all users automatically belongs to them. This means that there is no system management required for systems that do not need privilege checking.

Setting RTR Privileges on OpenVMS Systems

Create the Rights Identifiers RTR$OPERATOR and RTR$INFO if they do not already exist on your system, and assign them to users as appropriate. The RTR System Manager must have the RTR$OPERATOR identifier or the OPER privilege.

Setting RTR Privileges on Windows NT Systems

Administrator privileges are needed for RtrOperator rights by the RTR System Manager.

2.8 RTR ACP Virtual Memory Sizing

In addition to basic memory requirements of an unconfigured RTR ACP of approximately 5.8 Mbytes, additional memory requirements may be required according to the operating system environment that the RTR ACP process is using.

Compaq strongly recommends that you allocate as much virtual memory as possible. While there is no penalty for allocating more virtual memory than is used, the result of allocating too little can be catastrophic.

2.8.1 OpenVMS Virtual Memory Sizing

On OpenVMS the following allowances for additional virtual memory should be made:

- For each link add an additional 202 Kbytes
- For each facility an additional 13 Kbytes plus 80 bytes for each link in the facility
- For each client or server application process an additional 190 Kbytes for the first channel
- For each additional application channel an additional 1350 bytes

It is also necessary to prepare for the number of active transactions in the system. Unless the client applications are programmed to initiate multiple concurrent transactions this number will not exceed the total number of client channels in the system. This should be verified with the application provider.

In addition it is necessary to determine the size of the transaction messages in use:

1. For each front end:
    - Add one Kbyte per active transaction
    - Add 250 bytes per message per transaction

- Add the size of all messages

2. For each transaction router:
   - Allow one Kbyte for each active transaction
3. For each back end:
   - Allow one Kbyte per active transaction
   - Allow fifty bytes for each message of a transaction
   - Add the size of all replies

The total of all the contributions listed will yield an estimate of the likely virtual memory requirements of the RTR ACP. A generous additional safety factor should be applied as a final element to the total of virtual memory sizing. It is better to grant the RTR ACP resource limits exceeding its real requirements than to risk loss of service in a production environment as a result of insufficient resource allocation. The total result should be divided by the virtual memory size in pages to obtain the final virtual memory requirement. Process memory and page file quotas should be set to accommodate as least this much memory.

Process quotas are controlled by qualifiers to the START RTR command. START RTR accepts both links and application processes as qualifiers which can be used to specify the expected number of links and application processes in the configuration. The values supplied are used to calculate reasonable and safe minimum values for the following RTR ACP process quotas:

- ASTLM
- BIOLM
- DIOLM
- FILLM
- PGFLQUOTA

Both /LINKS and /PROCESSES have high default values:

- The default value for /LINKS is now 512—This value is high but is chosen to protect RTR routers against a failover where the number of front ends is large and the number of surviving routers is small. The maximum value for /LINKS is 1200 which is unchanged from earlier versions of RTR on OpenVMS.
- The default value for /PROCESSES is 64— This value is large for front end and router nodes but is sized for back ends hosting applications. Back ends with complex applications may have to set this higher. The maximum value for /PROCESSES is the OpenVMS allowed maximum. Warning messages are generated if the requested (or default) memory quotas conflict with the system-wide WSMAX parameter, or if the calculated or specified page file quota is greater than the remaining free page file space.
- The default values for /LINK and /PROCESSES require a large page file. RTR issues a warning if insufficient free space remains in the page file to accommodate RTR, so choose values appropriate for your configuration.

Use of /LINK and /PROCESSES do not take into account memory requirements for transactions. If an application passes a large amount of data from client to server or vice-versa this should be included in the sizing calculations. For further information on the START RTR qualifiers see the START RTR command in the Command Reference section.

Once the requirements have been determined for the START RTR qualifiers of /PGFLQUOTA or /LINK and /PROCESSES then RTR should be started with these qualifiers set to ensure the appropriate virtual memory quotas are set.

Note

The AUTHORIZE utility of OpenVMS does not play a role in the determination of RTR ACP quotas. RTR uses AUTHORIZE quotas for the command line interface and communication server, COMSERV. Virtual memory sizing for the RTR ACP are determined through the qualifiers of the START RTR command.

2.8.2 UNIX Virtual Memory Sizing

The RTR ACP process requires the operator to size the process limits for the ACP before starting RTR on all platforms. No direct control of the process quotas of the RTR ACP is offered for UNIX based platforms but log file entries will result if hard limits are found to be less than the preferred values for the RTR ACP.

The following are the minimum limits for the ACP on the following UNIX platforms:

- On Compaq Tru64 UNIX:
  - A minimum of 1024 open file descriptors
  - A minimum of 1073742 Kbytes for virtual memory address space
  - A minimum of 268436 Kbytes for a single file size
  - A minimum of 419430 Kbytes for heap data segment sizing
  - A minimum of 33555 Kbytes for core file size
  - A minimum of 8389 Kbytes for stack segment size
  - A minimum of 0 for CPU time
- On Sun Solaris:
  - A minimum of 256 open file descriptors
  - A minimum of 1073742 Kbytes for virtual memory address space
  - A minimum of 268436 Kbytes for a single file size
  - A minimum of 419430 Kbytes for heap data segment sizing
  - A minimum of 33555 Kbytes for core file size
  - A minimum of 8389 Kbytes for stack segment size
  - A minimum of 0 for CPU time
- On IBM AIX:
  - A minimum of 268436 Kbytes for a single file size

  - A minimum of 419430 Kbytes for heap data segment sizing
  - A minimum of 33555 Kbytes for core file size
  - A minimum of 8389 Kbytes for stack segment size
  - A minimum of 0 for CPU time
- On HPUX:
  - A minimum of 1024 open file descriptors

The START RTR qualifiers /LINK and /PROCESSES apply only to the OpenVMS platform and the determination of process quotas on UNIX platforms must be done through operating system handling of virtual memory sizing.

2.9 Network Transports

RTR supports multiple network transports with a default behaviour as follows:

If an attempt to create a network connection to a remote node fails, RTR retries the connection attempt using an alternate transport protocol if one is available. The order in which the supported transport protocols are used depends on the host operating system:

- OpenVMS - first DECnet then TCP/IP.
- All other platforms - first TCP/IP, then DECnet.

If a connection attempt fails on all available protocols, the connect fails and is retried at a later time starting again with the first transport protocol.

If an established link fails, RTR automatically initiates a reconnection of the link, starting with the first transport protocol for the platform, regardless of the transport employed when the link failed.

2.9.1 Specifying the Link Transport Protocol

It is possible to override the protocol failover mechanism by specifying the transport protocol to be employed for a link by naming links to include a transport selecting prefix. Prefixing links names with "tcp." and "dna." specifies TCP/IP or DECnet as the required transports respectively. Use of these prefixes causes the local node to employ only the specified transport protocol when attempting a connection on the link to which the prefix has been applied. Note that use of a protocol prefix on one node does not prevent a remote node from connecting using some other transport.

For example, to specify the facility “FAX1” that only uses the DECnet transport, two routers (“DNET1” and “DNET2”), two backends (“SRV1” and “SRV2”) and many frontends, use the following command:-

```
RTR> create facility FAX1 /frontend=(dna.FE1,dna.FE2,dna.FE3 ....)
    /router=(dna.DNET1,dna.DNET2)
    /backend=(dna.SRV1,dna.SRV2)
```

Creating a facility that uses only TCP/IP would use a command like this:-

```
RTR> create facility FINANCE /frontend=(tcp.client1,tcp.client2,tcp.client7 ....)
    /router=(tcp.routr1,tcp.routr2)
    /backend=(tcp.srv1,tcp.srv2)
```

2.9.2 Using RTR with DHCP and Internet Tunnels

When using RTR with DHCP or an Internet tunnel, a nodename may not be fully known; special naming techniques are provided for these conditions.

Anonymous Clients

RTR allows the use of wild cards when specifying the frontends that a router is permitted to accept connections from (that is, in the facility definition on the router). Valid wild card characters are "*", "%" and "?". The result of using a wild card character at facility configuration time is the creation of a template link.

When operating RTR in conjunction with the Compaq Internet Personal Tunnel, a client system outside of the corporate firewall uses tunnel software to obtain a secure channel from the Internet to inside the corporate domain. The tunnel client is assigned an address by the tunnel server from a pool when the tunnel software starts up.

When an RTR router receives a connection request from RTR running on this client, the source of the address is the address assigned by the tunnel server. There is no longer a fixed relationship between the client and its address. The method of configuring the router to accept such a connection is to define the frontends nodes with all the possible addresses that the tunnel server can assign to tunnel clients; you can do this with wildcards. For example,

```
RTR> create facility . . ./frontend=*.pool.places.dec.com
```

This command enables all nodes connecting through the tunnel to connect as frontends. The anonymous client feature may also be used with frontends that are using DHCP for TCP/IP address assignment.

Using the Tunnel Prefix

By using the node name prefix "tunnel.", it is possible to configure RTR to accept a network connection from a particular remote node even if it is connecting via a Internet tunnel using an unknown pseudoadapter address. This method allows stricter access control than the anonymous client feature where wild cards may be used when specifying a remote node name. For example, on the router node behind a firewall, the facility definition could include:

```
RTR> create facility . . ./router=router.rtr.dec.com -
    /frontend=tunnel.client.rtr.dec.com
```

The definition on the frontend could be

```
RTR> create facility /router=router.rtr.dec.com -
    /frontend=client.rtr.dec.com
```

Troubleshooting Tunnel and Wildcard Connections

To assist in diagnosing connect acceptance problems, use the monitor picture ACCFAIL. This picture displays the recent history of those links from which the local node has refused to accept connections. It displays the failed link name as provided by the network transport, and can assist in rapidly identifying any problems.

TCP Services File

- RTR uses the TCP/IP port number 46000 for the network communication daemon `rtr rtrd`.

  On UNIX platforms, you should edit the file `/etc/services` to add the line

  ```
  rtracp          46000/tcp
  ```

This informs the system administrator that port number 46000/tcp is reserved for RTR. (Note that the RTR daemon is started by RTRACP and not by `inetd`).

2.9.3 Interoperation with RTR Version 2 Using DECnet

Reliable Transaction Router is interoperable with RTR Version 2.2D ECO3 or later when running on a platform that supports DECnet; that is OpenVMS, Compaq Tru64 UNIX, SUN, Windows 95 or Windows NT.

Note that it is not possible to mix Version 2 and Version 3 routers and backends; all router and backend nodes in a facility must be either Version 2 or Version 3. Frontend nodes may be either Version 2 and Version 3.

Defining the facility:

- On RTR Version 2 node(s):- There are no special requirements for including a V3.x frontend in a V2 facility definition. Simply add the name of the frontend to the node-list specified by the /FRONTEND qualifier.
- On RTR Version 3 node(s):-
  The default network transport for RTR Version 3.2 is TCP/IP. [1] Since RTR Version 2 uses DECnet (only), you must specify that your RTR Version 3 nodes use the DECnet protocol. This is achieved by prefixing the RTR V2 nodename with the string "dna.".

  For example, to specify the facility "FAX1" on the RTR V3 frontend "v3fe" for which the two V2 routers "VMS1" and "VMS2" are defined, use the following command:-

  ```
  RTR> create facility FAX1 /frontend=v3fe /router=(dna.VMS1,dna.VMS2)
  ```
- Note that the facility name should contain only UPPER-CASE letters on all nodes if the facility includes nodes running RTR V2.

The use of the "dna." prefix assumes that the default network transport is TCP/IP. The default network transport can be changed to DECnet by setting the environment variable RTR_PREF_PROT. On Windows 95 and Windows NT, you can use one of the following statements in your `AUTOEXEC.BAT`.

```
set RTR_PREF_PROT=RTR_TCP_FIRST
set RTR_PREF_PROT=RTR_TCP_ONLY
set RTR_PREF_PROT=RTR_DNA_FIRST
set RTR_PREF_PROT=RTR_DNA_ONLY
```

These set the choice of network transport to TCP/IP with fallback to DECnet, TCP/IP only, DECnet with fallback to TCP/IP or DECnet only.

For Reliable Transaction Router Version 3.2 for OpenVMS, refer to Section 2.10 for further information))

Trouble-shooting network connections:-

If the RTR V3 frontend fails to connect with the RTR V2 router node, then you can make a basic check by executing a `dlogin` from the RTR V3 node to the OpenVMS router node. If this fails, consult your Network Manager. (For Compaq Tru64 UNIX machines, ensure that the DECnet library is installed as `/usr/shlib/libdna.so`).

[1] For Reliable Transaction Router Version 3.2 for OpenVMS, the default network transport is DECnet.

2.10 Network Protocol Selection on OpenVMS

- The default network transport protocol on OpenVMS is DECnet. You may change the default to TCP/IP by removing this line from RTR$STARTUP.COM:

  ```
  $ DEFINE/SYSTEM RTR_PREF_PROT RTR_DNA_FIRST
  ```

  If you are using TCP/IP, you will need to use the node-name prefix "dna." if you specifically want DECnet transport to be used. This is required, for example, when connecting to Version 2.2D ECO6 nodes as described in Section 2.9.3 of these Notes, and Section 2.7 of the *System Manager's Manual*.

  If you are using DECnet as the default, you will need to use the node-name prefix "tcp." to connect to other nodes using TCP/IP transport.

  If the value of the logical RTR_PREF_PROT is changed, the new value takes effect only after RTR has been restarted.

- Reliable Transaction Router Version 3.2 for OpenVMS can use either Compaq TCP/IP Services for OpenVMS or TCPware Version 5.1 as the TCP/IP transport layer.

2.11 Running RTR as a Service on Windows NT

Once the RTR as Service has been installed (see *Installation Guide*), RTR may be started or stopped from the Control Panel / Services panel using the START and STOP buttons provided.

- To start RTR: Press the START button.
- To stop RTR: Press the STOP button.

Note

Pressing START and STOP or the reverse in quick succession (within five or so seconds, depending on the speed of your computer) may cause undesirable results. This is because the Service executes quickly, making available the other action button, but the requested RTR action may not have completed when the second action button is pressed. It is therefore possible, for example, that the STOP action may be blocked by an incomplete START action. Although the Service will claim to be STOPped, RTR may in fact remain started. Pressing whichever action button is functioning should repair the problem.

By default, RTR will not restart automatically at system reboot time. You can change this by setting the Control Panel / Services entry for RTR.

Occasionally, an RTR process may continue to run after STOP has been pressed, and subsequent START and STOP actions may have no effect or produce an error. Under these circumstances, it will be necessary to intervene directly, as a privileged (SYSTEM) user, to stop RTR. This can be done either using RTR commands or with the Task Manager, or by rebooting.

2.11.1 Customizing the RTR Windows NT Service

While starting RTR, the Service looks for the file UsrStart.RTR in the RTR home directory. On finding the file, the Service executes any RTR commands it may contain. RTR commands from UsrStart.RTR execute after RTR has been started.

From the point of view of the Service, the RTR home directory is found in the system-level environment variable RTR_DIRECTORY, or, if that is not defined, then the directory from which the Service was executed.

For the RTR Service to use it, RTR_DIRECTORY must be defined in the system-level environment variables list, not the user-level environment variables list. Also, the system must be rebooted after the definition of RTR_DIRECTORY is either created or changed for it to be used.

If a user-level copy of RTR_DIRECTORY exists, it must identify the same RTR home directory as the system-level copy, or if there is no system-level copy, the directory containing the currently registered Service program. If it does not, behavior of RTR is undefined. Changing the value of RTR_DIRECTORY, or reregistering the service from another directory while RTR is running, is dangerous and should be avoided. Starting RTR from the Service, then stopping it from DOS (or the reverse) should also be avoided.

If you put STOP RTR in the UsrStart.RTR file, it will stop RTR. The Service will not detect that RTR has been stopped and will offer only the STOP action button. Pressing the STOP button will fix the problem.

Similarly, when the Service stops RTR, it searches the RTR home directory for the file UsrStop.RTR and, if the file exists, execute any RTR commands in it. User commands from UsrStop.RTR are executed before RTR has stopped.

WARNING

If you put QUIT or EXIT in either UsrStart.RTR or UsrStop.RTR, RTR will exit improperly. As a result, an RTR command server process incorrectly remains active, preventing the Service from starting or stopping RTR, and preventing the RTR command server from exiting. Because the RTR command server executes under the SYSTEM account, it cannot be stopped from Task Manager other than by the SYSTEM account.

2.11.2 Files Created by the RTR Windows NT Service

If RTR is started from the Service rather than via a Command Prompt window, several files are created in the RTR root directory. SrvcIn.Txt is created to act as a command line input source; SrvcOut.Txt acts as a container for console output; RTRStart.RTR contains the startup commands. When the Service stops RTR, it recreates SrvcIn.Txt and creates RTRStop.RTR for stopdown commands. Creation of these files is unconditional; that is, they are created every time RTR is started or stopped, whether or not they already exist. RTR will thus ignore (and overwrite) any changes made to one of these files.

2.12 How RTR Selects Processing-states (Roles) for Nodes

This section discusses how RTR assigns roles to backend node partitions, and how routers are selected.

2.12.1 Role Assignment for Backend Node Partitions

RTR assigns a primary or secondary processing state to a partition (or a key-range definition), consisting of one or more server application channels, which may or may not share a common process. All such server channels belonging to a given partition will have the same processing state on a given node. However, the processing state for the same partition will normally be different on different nodes. The exception is the case of the standby processing state. Because a given partition can have multiple standby nodes, several of these nodes may be in this state.

RTR determines the processing state of a given partition through the use of a globally managed sequence number for that partition. By default, the RTR master router will automatically assign sequence numbers to partitions during startup. When a server is started up on a backend node and declares a new partition for that node, the partition initially has a sequence number of zero. When the partition on that backend makes an initial connection to the master router, the router increases its sequence number count for that partition by one and assigns the new sequence number to the new backend partition. The active node with the lowest backend partition sequence number gets the primary processing state in both shadow and standby configurations. That node is also referred to as the primary node, though the same node could have a standby processing state for a different partition.

Under certain failover conditions, backend partitions may either retain their original sequence number or be assigned a new sequence number by the router. If a failure is caused by a network disruption, for example, a backend partition will retain its sequence number when it reconnects with the router. However, if the backend node is rebooted or RTR is restarted on the backend node, a new sequence number will be assigned by the router to any partitions that start up on that node. Routers will only assign new sequence numbers to backend partitions that have a current sequence number of zero, or if the backend partition is joining an existing facility and has a sequence number that conflicts with another backend partition on another node.

Sequence number information can be obtained from the SHOW PARTITION command. In the output of this command the sequence number is indicated by the relative priority. The following example shows a sample of the SHOW PARTITION command from a router partition. This example shows that the backend partition called Bronze has a sequence number of 1, and backend partition called Gold has a sequence number of 2.

```
Router partitions on node SILVER in group test at Mon Mar 22 14:51:16 1999

State:                      ACTIVE
Low bound:                       0     High bound:              4294967295
Failover policy:                                          fail_to_standby
Backends:                                                      bronze,gold
 States:                                                  pri_act,sec_act
 Relative priorities:                                               1,2
Primary main:               bronze     Shadow main:                 gold
```

The SHOW PARTITION command on each backend node is as follows:

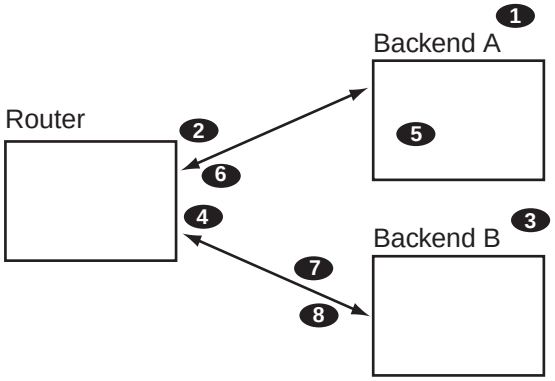
```
Backend partitions on node BRONZE in group "test" at Mon Mar 22 14:52:32 1999

Partition name:                                                        p1
Facility:       RTR$DEFAULT_FACILITY     State:                    pri_act
Low bound:                         0     High bound:            4294967295
Active servers:                    0     Free servers:                   1
Transaction presentation:     active     Last Rcvy BE:                gold
Active transaction count:          0     Transactions recovered:         0
Failover policy:     fail_to_standby     Key range ID:            16777217
Master router:                silver     Relative priority:              1
Features:                                       Shadow,NoStandby,Concurrent

Backend partitions on node GOLD in group "test" at Mon Mar 22 14:54:12 1999

Partition name:                                                        p1
Facility:       RTR$DEFAULT_FACILITY     State:                    sec_act
Low bound:                         0     High bound:            4294967295
Active servers:                    0     Free servers:                   1
Transaction presentation:     active     Last Rcvy BE:              bronze
Active transaction count:          0     Transactions recovered:         0
Failover policy:     fail_to_standby     Key range ID:            16777216
Master router:                silver     Relative priority:              2
Features:                                       Shadow,NoStandby,Concurrent
```

The following description shows how sequence numbers are initially assigned in a simple partition with two backends named Bronze and Gold, and a router named Silver.



1. A partition (with shadowing enabled) is started on node Bronze.
2. The partition on Bronze obtains sequence number 1 from the router and becomes the primary.
3. Another server on the same partition (with the same attributes) is started on Gold.
4. The partition on Gold obtains sequence number 2 from the router and becomes the secondary.
5. Node Bronze crashes and reboots (the partition sequence number on Bronze is reset to 0). The partition on Gold goes into Remember.
6. When the server starts, The partition on Bronze obtains sequence number 3 from the router and becomes the secondary, Gold now becomes the primary.
7. The network connection from node Silver to node Gold fails. The partition on Bronze becomes the primary. The partition on node Gold loses quorum and is in a wait-for-quorum state.

8. The network connection to node Gold is reestablished. The partition on Gold retained its original sequence number of 2 and retains the primary role while the partition on Bronze reassumes the secondary role.

Alternately, the roles of backend nodes can be specifically assigned with the /PRIORITY_LIST qualifier to the SET PARTITION command. In the previous example the /PRIORITY_LIST qualifier can be used to insure that when Bronze fails and then returns to participate in the facility it then becomes the active primary member. To insure this, the following command would be issued on both backend systems immediately after the creation of the partition:

```
SET PARTITION test/PRIORITY_LIST=(bronze,gold)
```

It is recommended that the same priority list order be used on all partition members. If a different list is used then the router will determine the sequence number for conflicting members through the order in which those members joined the facility. For example, if the above command were issued only on Bronze and Gold had the opposite priority list, then the router would assign the lower sequence number to the backend that joined the facility first.

The /PRIORITY_LIST feature is very useful in cluster configurations. For example, Site A and Site B each contain 2-node clusters. The facility is configured such that at Site A, Node-A1 has the primary active partition and Node-A2 has the standby partition. At Site B, Node-B1 is the secondary active partition and Node-B2 has the standby of the secondary. The partition could be defined such that the standby node, Node-A2, would become active if the primary node were to fail. For example, issuing the following command on all four nodes for this partition guarantees that the specified list is followed when there is a failure.

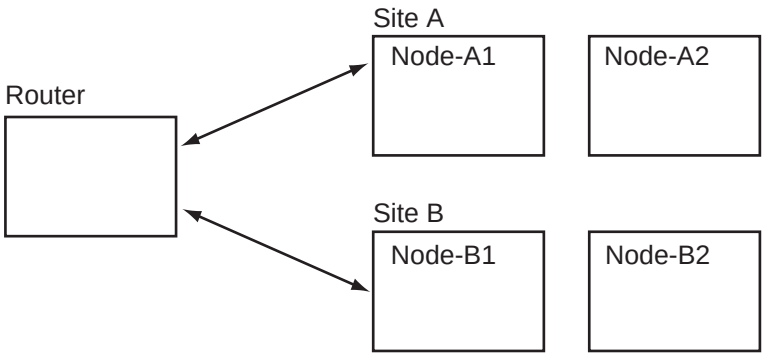
```
SET PARTITION test/PRIORITY_LIST=(Node-A1,Node-A2,Node-B1,Node-B2)
```

Using the SHOW PARTITION command from the router, this partition would be as follows:

```
Router partitions on node SILVER in group "test" at Mon Mar 22 17:22:06 1999

State:                        ACTIVE
Low bound:                         0     High bound:              4294967295
Failover policy:                                              fail_to_standby
Backends:                                    node-a1,node-a2,node-b1,node-b2
 States:                                     pri_act,standby,sec_act,standby
 Relative priorities:                                                1,2,3,4
Primary main:                 node-a1     Shadow main:                node-b1
```

However, the partition could also be configured so that the secondary active node, Node-B1, would become the primary node if the original primary system were to fail. This is controlled with the /FAILOVER_POLICY qualifier to the SET PARTITION command. The default is /FAILOVER_POLICY=STAND_BY.



If the relative priority (sequence number) for Node-A2 is changed to four it still becomes the primary active server if Node-A1 fails because the failover policy indicates a fail_to_standby requirement for this facility.

```
SET PARTITION test/PRIORITY_LIST=(Node-A1,Node-B1,Node-B2,Node-A2)
```

After issuing this command the router partition appears as follows. Note the change in relative priorities for the backends.

```
Router partitions on node SILVER in group test at Tue Mar 23 13:29:41 1999

State:                        ACTIVE
Low bound:                         0     High bound:              4294967295
Failover policy:                                                fail_to_standby
Backends:                                      node-a1,node-a2,node-b1,node-b2
 States:                                       pri_act,standby,sec_act,standby
 Relative priorities:                                                  1,4,2,3
Primary main:                 node-a1     Shadow main:                 node-b1
```

The following SET PARTITION command can be issued to change the facility so that Node-B1 will become the primary active server if Node-A1 fails.

```
SET PARTITION test/FAILOVER_POLICY=shadow
```

The /FAILOVER_POLICY qualifier is intended for use in selecting a new active primary in configurations where shadowing is enabled. This qualifier takes precedence over the /PRIORITY_LIST qualifier. The /PRIORITY_LIST qualifier is intended for use in determining the failover order for specific nodes. It is most useful in cluster configurations where it can be used to specify the exact failover order for the nodes within the cluster. For example, in a standby facility on a cluster of four nodes, the /PRIORITY_LIST qualifier can be used to specify the desired order of failover for those cluster members. Some machines within a cluster may be more powerful than other machines. This feature allows for the most efficient use of those machines.

2.12.2 Router Selection

Within the scope of a given facility, routers and backends connect to one another. However, nodes with a specific role do not connect to nodes with the same role, i.e., routers do not connect to other routers. Frontends choose only one router to connect to at a given time. This router is called the Current Router for that frontend within the scope of a facility.

A backend connects to all routers defined within a facility. The connected router with the lowest network address is designated the master router. Internally, a node is identified through a structure called the Kernel Net ID. The Kernel Net ID is a concatenation of all network addresses a node is known as for all the protocols and interfaces that it supports. The master router designation is only

relevant to a backend. It is where the backend goes to obtain and verify partition configuration and facility information.

Routers are made known to the frontend systems through the list specified in the /ROUTER=(list) qualifier to the CREATE FACILITY command. This list specifically determines the preferred router. If the first router specified is not available, the next one on the list is chosen. When the facility is created on the frontend, the list of routers specified can be a subset of the routers contained within the entire facility. This can be used to prevent a frontend from selecting a router that is reserved for other frontend systems. Fail back of routers is supported. This means that if the preferred router was not available, and it became available later, the frontend would automatically fail back and connect to its preferred router.

Router connectivity can also be controlled through the use of the /BALANCE qualifier either on the CREATE FACILITY command or on the SET FACILITY command. When the /BALANCE qualifier is used, the list of routers specified in the router list is randomized, making the preferred router a random selection within the list. Assume the following command is issued from a frontend:

```
RTR CREATE FACILITY test/FRONTEND=Z/ROUTER=(A,B,C)
```

The frontend attempts to select a router based on the priority list A, B, C, with A being the preferred router. If the /BALANCE qualifier is added to the end of this command then the preferred router is randomly selected from the three nodes. This random list exists for the duration of the facility. After the facility is stopped, a new random list is made when the facility is created again. The exception to this is if a router does not have quorum (sufficient access to backend systems) then that router will no longer accept connections from frontend systems until it has again achieved quorum. The /BALANCE qualifier is only valid for frontend systems.

3
Partition Management

3.1 Overview

This section describes the concepts and operations of RTR's partitions.

3.1.1 What is a Partition?

Partitions are subdivisions of a routing key range of values. They are used with a partitioned data model and RTR data content routing. Partitions exist for each distinct range of values in the routing key for which a server is available to process transactions. RTR provides for failure tolerance by allowing system operators to start redundant instances of partitions in a distributed network and by automatically managing the state and flow of transactions to the partition instances.

Partition instances support the following relationships:

- Concurrency - this attribute permits multiple server channels to be connected to an instance of a partition.
- Standbys - multiple instances of a partition distributed over the nodes of a cluster. A standby set may have as many members as a cluster has nodes, or with some restrictions you may place a standby on any network node. At any one time, one member of the set is active while the others wait in standby mode to take over in the event of failure of the active member.
- Shadows - shadow instances provide site disaster protection by allowing replication of transaction processing at a remote site. A pair of partition instances or (standby sets thereof) cooperate to provide this replication, with provision for automatic recovery of a shadow member restarting after a failure.

Prior to RTR V3.2, the creation and behavior of a partition was tied to the declaration of server application channels. Partitions and their characteristics can now be defined by the system operator. This has the following advantages:

- It allows a further de-coupling of the application from its operating environment, thus reducing application programming requirements.
- Allows the system operators to make choices concerning the runtime behavior of the system.

3.1.2 What is Partition Management?

Before RTR V3.2, the management of the state of a partition was an entirely automatic function of the distributed RTR system. Starting with RTR V3.2, the system operator can issue commands to control certain partition characteristics, and to set preferences concerning partition behavior.

3.2 Partition Naming

A prerequisite for partition management is the ability to identify a partition in the system that is to be the subject of management commands. For this purpose, partitions have been given names, which may be drawn from a number of sources described below.

3.2.1 Default Partition Names

Unless supplied by one of the methods described below, partitions receive automatically generated default names. They allow system operators access to the partition command set without the need to change existing application programs or site configuration procedures.

3.2.2 Programmer Supplied Names

An extension to the `rtr_open_channel( )` call allows the application programmer to supply a name when opening a server channel. The `pkeyseg` argument specifies an additional item of type `rtr_keyseg_t`, assigning the following values:

- `ks_type = rtr_keyseg_partition`, indicating that a partition name is being passed.
- code_example>(ks_lo_bound) should point to the null-terminated string to use for the partition name.
- code_example>(ks_hi_bound) must be NULL.

Using this model, the partition segments and key ranges served by the server are still specified by the server when the channel is opened.

3.2.3 System Manager Supplied Partition Names

Partitions can be defined by the system manager through the use of the code_ example>(create partition) system management command, or through use of `rtr_open_channel( )` flag arguments. The system manager can set partition characteristics with this command and applications can open channels to the partition by name. See the Section 3.4 for an example of passing a partition name with `rtr_open_channel( )`.

3.2.4 Name Format and Scope

A valid partition name must contain no more than 63 characters in length and can combine alphanumeric characters (abc123), the plus sign (+), the underscore (_), and the dollar sign ($). Partition names must be unique within a facility name and should be referenced on the command line with the facility name when using partition commands. Partition names exist only on the backend where the partition resides. You won't see the partition names at the RTR routers.

3.3 Life Cycle of a Partition

3.3.1 Implicit Partition Creation

Partitions are created implicitly when an application program calls rtr_open_channel( ) to create a server channel, specifying the key segments and value ranges for the segments with the pkeyseg argument. Other partition attributes are established with the flags argument. Before RTR V3.2, this was the only way in which partitions could be created. Partitions created in this way are automatically deleted when the last server channel to the partition is closed.

3.3.2 Explicit Partition Creation

Partitions can also be created by the system operator before server application program start up using system management commands. This gives the operator more control over partition characteristics. Partitions created in this way remain in the system until either explicitly deleted by the operator, or RTR is stopped.

3.3.3 Persistence of Partition Definitions

RTR stores partition definitions in the journal, and records for each transaction the partition in which it was processed. This is convenient when viewing or editing the contents of the journal, where the partition name can be used to select a subset of the transactions in the journal. RTR will not permit a change in the partition name or definition as long as transactions remain in the journal that were processed under the current name or definition for the partition. If transactions remain in the journal and you need to change the partition name or definition, you can take the following actions:

- Start appropriate servers to complete processing of the transactions.
- Remove the transactions from the journal with the SET TRANSACTION command.
- Replace the RTR journal with the CREATE JOURNAL/SUPERSEDE command. Note that this will destroy any transactions remaining in the journal and should be done with caution.

3.4 Binding Server Channels to Named Partitions

For a server application to be able to open a channel to an explicitly created partition, the application passes the name of the partition through the pkeyseg argument of rtr_open_channel( ) call. It is not necessary to pass key segment descriptors, but if the application does so, they must be compatible with the existing partition definition. You may pass partition characteristics through the flags argument, but these will be superseded by those of the existing partition.

Example:

```
RTR> create partition/KEY1=(type. . .) par_one
 . . .
rtr_keyseg_t    partition_name;

partition_name.ks_type = rtr_keyseg_partition;
partition_name.ks_lo_bound = "par_one";

status - rtr_open_channel(  . . ., RTR_F_OPE_SERVER, . . ., 1, &partition_name);
```

Summarizing, to fully de-couple server applications from the definition of the partitions to be processed, write applications that open server channels where only the required partition name is passed. Leave the management of the partition characteristics to the system managers and operators.

3.5 Entering Partition Commands

Partitions can be managed by issuing partition commands directed at the required partition after they are created. Partition commands can be entered in one of two ways:

- A command line processed by the RTR command line interface, for example `RTR> SET PARTITION`
- Programmed using `rtr_set_info( )`

Enter partition commands on the backend where the partition is located. Note that commands that affect a partition state only take effect once the first server joins a partition. Errors encountered at that time will appear as log file entries. Using partition commands to change the state of the system causes a log file entry.

3.5.1 Command Line Usage

Partition management in the RTR command language is implemented with the following command set:

- `RTR> CREATE PARTITION`
- `RTR> SET PARTITION`
- `RTR> DELETE PARTITION`

The name of the facility in which the partition resides may be specified with the **/FACILITY** command line qualifier, or as a colon-separated prefix to the partition name (for example Facility1:Partition1). Detailed descriptions of the command syntax are given in the Command Line Reference section of this manual, and are summarized in the discussions below. Examples in the following sections use a partition name of Partition1 in the facility name of Facility1.

3.5.2 Programmed Partition Management

Partition commands are programmed using `rtr_set_info( )`. Usage of the arguments are as follows:

- `pchannel` - Supplies the address of a `rtr_channel_t` to receive the channel opened in the event of a successful call.
- Flags must be `RTR_NO_FLAGS`
- Verb must be the value `verb_set` (from the enumeration rtr_verb_t)
- Object must be `rtr_partition_object`
- `select_qualifiers` should identify the facility and partition, by name, for example:

```
rtr_qualifier_value_t select_qualifiers[ 3 ];
select_qualifiers[ 0 ].qv_qualifier = rtr_facility_name;
select_qualifiers[ 0 ].qv_value = "your_facility_name_here";
select_qualifiers[ 1 ].qv_qualifier = rtr_partition_name;
select_qualifiers[ 1 ].qv_value = "your_partition_name_here";
select_qualifiers[ 2 ].qv_qualifier = rtr_qualifiers_end;
select_qualifiers[ 2 ].qv_value = NULL;
```

- The `set_qualifier` list expresses the required change in partition behaviour or characteristic.

The rtr_set_info( ) call completes asynchronously. If the function call is successful, completion will be signaled by the delivery of an RTR message of type rtr_mt_closed on the channel whose identifier is returned through the pchannel argument. The programmer should retrieve this message by using rtr_receive_message( ). The data accompanying the message is of type rtr_status_data_t. The completion status of the partition command can be accessed as the status field of the message data.

3.6 Managing Partitions

To manage partitions a set of commands or program calls are used. Information on managing partitions is provided in this section.

3.6.1 Controlling Shadowing

The state of shadowing for a partition can be enabled or disabled. This may be useful in the following circumstances:

- Enabling site disaster protection for an application partition for the first time
- A recovery aid following prolonged outage of a former shadow site.

The following restrictions apply. Shadowing for a partition can be turned off only in the absence of an active secondary site, The active member must be running in remember mode. The command will fail if entered on either an active primary or secondary with a message to this effect. If entered on a standby of either the primary or secondary, the command is accepted but fails in the RTR router. This failure is recorded with a log file entry at the router. Once shadowing is disabled, the secondary site servers will be unable to startup in shadow mode until shadowing is enabled again. Shadowing for the partition can be turned on by entering the command at the current active member or on any of its standbys.

3.6.1.1 Command Line Example

```
RTR> SET PARTITION/FACILITY=Facility1/SHADOW Facility1:Partition1
```

For further information see the SET PARTITION command in Chapter 6.

3.6.1.2 Programming Information

To enable shadowing, program the set_qualifier argument of rtr_set_info( ) as follows:

```
rtr_qualifier_value_t   set_qualifiers[ 2 ];
rtr_partition_state_t  newState = rtr_partition_state_shadow;

set_qualifiers[ 0 ].qv_qualifier = rtr_partition_state;
set_qualifiers[ 0 ].qv_value  = &newState;
set_qualifiers[ 1 ].qv_qualifier = rtr_qualifiers_end;
set_qualifiers[ 1 ].qv_value  = NULL;
```

To disable shadowing, specify newState as rtr_partition_state_noshadow.

3.6.2 Controlling Transaction Presentation

Transaction presentation is the process of passing transactions to idle server channels for processing. While transaction presentation is active, new transactions are started on the first free server channel for the appropriate partition.

Use the /SUSPEND qualifier to the SET PARTITION command to halt the presentation of new transactions to servers on the backend where the command is entered. The command completes when the processing of all currently active transactions is complete. The optional /TIMEOUT qualifier specifies, as a number of seconds, the time that the command waits for completion. If the command times out, presentation of new transactions are suspended, but there still exist transactions for which servers have yet to complete processing. The operator must decide either to reenter the command and wait a further period of time, or resume the partition. Note that use of this command does not affect any transaction timeout value specified by RTR clients, so such transactions may encounter a timeout condition if the partition remains suspended.

/RESUME qualifier restarts presentation of transactions to the server application channels.

3.6.2.1 Command Line Example

Example usage of the qualifiers:

```
RTR> SET PARTITION/FACILITY=Facility1/SUSPEND/TIMEOUT=5 Facility1:Partition1
RTR>
RTR> SET PARTITION/FACILITY=Facility1/RESUME Facility1:Partition1
```

For a more complete description see the SET PARTITION command in Chapter 6.

3.6.2.2 Programming Information

To suspend transaction presentation on a partition with a timeout of 30 seconds, program the set_qualifier argument of the rtr_set_info( ) call as follows:

```
rtr_qualifier_value_t   set_qualifiers[ 3 ];
rtr_partition_state_t  newState = rtr_partition_state_suspend;
rtr_uns_32_t   ulTimeoutSecs = 30;

set_qualifiers[ 0 ].qv_qualifier = rtr_partition_state;
set_qualifiers[ 0 ].qv_value  = &newState;
set_qualifiers[ 1 ].qv_qualifier = rtr_partition_cmd_timeout_secs;
set_qualifiers[ 1 ].qv_value  = &ulTimeoutSecs;
set_qualifiers[ 2 ].qv_qualifier = rtr_qualifiers_end;
set_qualifiers[ 2 ].qv_value  = NULL;
```

Note that the timeout is an optional element. To resume transaction presentation, specify newState as rtr_partition_state_resume.

3.6.3 Controlling Recovery

The purpose of RTR automated recovery is to ensure the best possible consistency of application databases across a distributed computing environment. To achieve this RTR relies in part on information stored in the journals of the participating systems. Should one or more of these systems be unavailable at recovery time, automated recovery may stall or fail awaiting availability of these systems and their journals. This is good from the point of view of data consistency, but bad when viewed from an application availability perspective.

If a partition enters a wait state or fails but has neither a local or remote journal, an operator can instruct RTR to skip the current step in the recovery process with the /IGNORE_RECOVERY qualifier. Since this command bypasses parts of the recovery cycle use it with caution in cases where availability above consistency in application databases is desired.

The recovery cycle can also be manually restarted with the /RESTART_RECOVERY qualifier. This may be useful if the operator previously aborted automated recovery. Since this command can result in recovery of transactions from previously inaccessible journals, do not use this if your applications are sensitive to the order in which transactions are processed by the servers.

3.6.3.1 Command Line Example

Example of the qualifiers:

```
RTR> SET PARTITION/FACILITY=Facility1/IGNORE_RECOVERY Facility1:Partition1
RTR>
RTR> SET PARTITION/FACILITY=Facility1/RESTART_RECOVERY Facility1:Partition1
```

A complete description of the qualifiers to the SET PARTITION command can be found in Chapter 6.

3.6.3.2 Programming Information

To terminate the current recovery state, program the set_qualifier argument of rtr_set_info( ) as follows:

```
rtr_qualifier_value_t   set_qualifiers[ 2 ];
rtr_partition_state_t  newState = rtr_partition_state_exitwait;

set_qualifiers[ 0 ].qv_qualifier = rtr_partition_state;
set_qualifiers[ 0 ].qv_value = &newState;
set_qualifiers[ 1 ].qv_qualifier = rtr_qualifiers_end;
set_qualifiers[ 1 ].qv_value = NULL;
```

To restart recovery, specify newState as rtr_partition_state_recover.

3.6.4 Controlling the Active Site

RTR lets the system operator to deploy a range of shadow and standby partitions in order to provide the desired degree of application resilience to failures. By default, RTR automatically manages the assignment of active and standby roles to the available partition instances. The operator can assign a relative priority to each backend on which a partition instance exists. Enter priority as a list of backend node names with the highest priority first in decreasing order. See the command example Section 3.6.4.1.

Suspend transaction presentation before entering or changing the priority list.

3.6.4.1 Command Line Example

```
RTR> SET PARTITION/PRIORITY_LIST=(BE1, BE2, BE3) Facility1:Partition1
```

For more information on the SET PARTITION command see Chapter 6.

3.6.4.2 Programming Information

To set the partition backend priority list, program the set_qualifier argument of the rtr_set_info( ) call as follows:

```
rtr_qualifier_value_t   set_qualifiers[ 2 ];
char    *szNodeList = "your,list,of,node,names,here"

set_qualifiers[ 0 ].qv_qualifier = rtr_partition_be_priority_list;
set_qualifiers[ 0 ].qv_value  = &szNodeList;
set_qualifiers[ 1 ].qv_qualifier = rtr_qualifiers_end;
set_qualifiers[ 1 ].qv_value  = NULL;
```

3.6.5 Controlling Failover

In a system configured for maximum fault tolerance employing both shadows and standbys, there is a choice to be made in case of the failure of the primary site. The qualifier to the SET PARTITION command of /FAILOVER_POLICY= allows the system operator to select one of the following policies that RTR should pursue in selecting the new primary site in the event of a failure:

- /FAILOVER_POLICY=STANDBY causes RTR to choose a standby of the failed primary (if any) to become the new primary. If there is more than one standby, the operator may addition use the priority list feature (described above) to control which standby is preferred. Depending on the size of the journal of the failed primary, there will be a hold up in the processing of transactions whilst the journal is recovered. This is the default behaviour.
- /FAILOVER_POLICY=SHADOW instructs RTR to make the active secondary (if any) the new primary. A standby of the failed primary (if any) will be elected to become the new secondary. This option gives the shortest fail over time, but will move the primary to a different cluster that you may have located at a different site.
- /FAILOVER_POLICY=COMPATIBLE_PRE_V32 is a mode that will operate with configurations that contain RTR routers running versions of the software prior to V3.2. This mode will be automatically adopted if such routers exists in or join the configuration.

3.6.5.1 Command Line Example

An example use of the /FAILOVER_POLICY qualifier:

```
RTR> SET PARTITION/FAILOVER_POLICY=SHADOW Facility1:Partition1
```

For more information see the SET PARTITION command in Chapter 6.

3.6.5.2 Programming Information

To set the partition failover policy, program the set_qualifier argument of the rtr_set_info( ) call as follows:

```
rtr_qualifier_value_t   set_qualifiers[ 2 ];
rtr_partition_failover_policy_t newPolicy;

set_qualifiers[ 0 ].qv_qualifier = rtr_partition_failover_policy;
set_qualifiers[ 0 ].qv_value = &newPolicy;
set_qualifiers[ 1 ].qv_qualifier = rtr_qualifiers_end;
set_qualifiers[ 1 ].qv_value = NULL;
```

Legal values for newPolicy are:

- rtr_partition_fail_to_standby

- rtr_partition_fail_to_shadow
- rtr_partition_pre32_compatible

3.6.6 Controlling Transaction Replay

RTR has implemented the capability of controlling transaction replay in cases where a "killer message" happens during a transaction replay preventing recovery from continuing normally. A "killer message" presents a situation where server availability is lost because of the presence of a message capable of causing repeated server application failure during recovery. This is typically the result of an improperly handled condition or application programming error within the server itself. Under such circumstances it may be desirable to sidestep a particular transaction, maintain server operation, and manually process the transaction at some later time.

The RTR solution is to establish, for a given partition, the maximum number of retries for any given transaction presented during recovery. Once this limit has been exceeded, the offending transaction is removed from the recovery process and is written to the journal as an exception record. Subsequent processing of this transaction requires manual intervention by someone qualified to evaluate and correct the situation in both the application and in RTR. Once the application status is understood, the `set transaction` command can be used to update the journal, thus insuring that the final state of any manually transacted exceptions are accurately reflected in future recovery operations.

The recovery retry count indicates the maximum number of times that a transaction should be presented for recovery before being written to the journal as an exception. Once a transaction has been recorded as an exception, it is no longer considered eligible for recovery and requires manual processing by a qualified individual.

The recovery retry count is partition-specific, and applies to both local and shadow recovery operations. The default is no limit on the number of retries, which permits a killer message to bring down all available servers servicing a given partition.

The recovery retry count should be set before starting (or restarting) the application servers so that the limit is established prior to the start of recovery operations.

3.6.6.1 Command Line Example

```
RTR> SET PARTITION/RECOVERY_RETRY_COUNT=3 Facility1:Partition1
```

For more information on the SET PARTITION command see Chapter 6.

3.6.6.2 Programming Information

To set the partition transaction recovery limit, program the `set_qualifier` argument of `rtr_set_info( )` as follows:

```
rtr_qualifier_value_t   set_qualifiers[ 2 ];
rtr_uns_32_t   newLimit = . . .;

set_qualifiers[ 0 ].qv_qualifier = rtr_partition_rcvy_retry_count;
set_qualifiers[ 0 ].qv_value = &newLimit;
set_qualifiers[ 1 ].qv_qualifier = rtr_qualifiers_end;
set_qualifiers[ 1 ].qv_value = NULL;
```

3.7 Displaying Partition Information

Information on the definition and state of a partition is displayed with the SHOW PARTITION command. The information of interest in the context of partition management relates to the backend instance of the partition. For more information see the SHOW PARTITION command in Chapter 6.

3.7.0.1 Command Line Example

```
RTR> show partition/backend

Backend partitions on node BE1 in group "Facility1" at Wed Feb 24 15:07:50 1999

Partition name                    Facility                      State
RTR$DEFAULT_PARTITION_16777217    RTR$DEFAULT_FACILITY          active
RTR$DEFAULT_PARTITION_16777218    RTR$DEFAULT_FACILITY          active
```

4
Transaction Management

4.1 Overview

This section describes the concepts of RTR's transaction management capability.

The RTR transaction is the heart of an RTR application, and transaction state is the property that characterizes a transaction's current condition. Whenever a transaction progresses from one stage to another, the transaction state is updated to reflect a transaction transition. Transaction states are maintained in memory and some types of transaction states are also stored in the RTR Journal for recovery purposes.

Three different types of states are used internally by RTR to keep track of transaction status.

- Transaction Runtime State
- Transaction Journal State
- Transaction Server State

These three state types are very closely related. The Transaction Runtime State, also known as Transaction State, describes how a transaction progresses from a RTR role (FE, TR, BE) point of view. For example, a transaction can enter a stage in which its transaction state from an RTR frontend viewpoint is different than the transaction state of an RTR router.

The Transaction Journal State describes how a transaction running on an RTR backend progresses from the RTR journal perspective. When a transaction transitions, its Transaction Journal State gets updated and the new state along with other information pertaining to this transaction is stored in the RTR journal. The Transaction Journal State is primarily used by RTR to perform the recovery replay of a transaction after a failure, if necessary. An RTR frontend and router will not see this state.

The Transaction Server State describes transaction state transition seen by the server. RTR uses this state to determine if a server is available to process a new transaction or if a server has voted on a particular transaction. As with the Transaction Journal State, the Transaction Server State is only managed at the backend.

RTR provides a set of comprehensive management utilities to help users closely monitor the flow of a transaction and all three types of states associated with that transaction. These utilities help users understand how a transaction migrates from one stage to another and help diagnose problems.

The RTR `SHOW TRANSACTION` command can be used to examine a transaction's up-to-date status on frontend, router or backend roles. With this command, users can see all three types of transaction states of a particular transaction and also understand how the RTR journal and application servers perceive this transaction. When a transaction commits or aborts, all status associated with

this transaction is removed from memory and can no longer be monitored by the command.

The RTR DUMP JOURNAL command can be used to trace and review the flow of a transaction. The RTR journal saves all of the information about a transaction, its transaction journal state, the transaction messages (records) received from the RTR client, and the content of a message sent to the server. The information will be kept until a transaction is committed and forgotten.

The RTR SET TRANSACTION command is used to modify a live transaction to change the current state of a transaction to a new state. This command can be used to circumvent a difficult situation. For example, in a situation where two shadowed servers are configured, the system administrator might decide not to replay (recover) all transactions in a shadowed RTR journal after a failure. The SET TRANSACTION command could set specified transactions in a PRI_DONE or remember state to a DONE state and avoid the delay of transactions being remembered from a journal for fast recovery. The SET TRANSACTION command should only be used by experienced RTR system administrators as the command introduces the risk of corrupting or losing transactions if used incorrectly. It can be used on the backend only and the RTR log file must be turned on for this command.

Log file entries are made for all transaction state changes for debugging and auditing purposes.

4.1.0.1 Command Line Examples

An example of the use of the SET TRANSACTION command:

```
RTR> start rtr
RTR> set log/file=settran
RTR> set transaction/state=PRI_DONE/new_state=DONE/facility=Facility1/partition=Partition1 *
```

This example would set all transactions with the wildcard * in the current state of PRI_DONE (remember) to DONE on the facility Facility1 and the partition Partition1. The log file, settran, would record the transaction state changes. The changes could be viewed with the SHOW TRANSACTION command or the DUMP JOURNAL command. In a shadow recovery situation this would clear the journal of remember transactions and provide for a fast recovery of access to the database if needed.

For detailed information on these commands see Chapter 6.

4.1.1 Exception Transactions

Transactions can cause servers to fail after the VOTE phase and impact availability of a server in a recovery. These "EXCEPTION" transactions can now be flagged by RTR as "fail transactions" after the user sets the attempts at recovery from a failure with the SET PARTITION/RECOVERY_RETRY_COUNT=nn command. They then can be identified and removed from the RTR journal and from the system to allow recovery to continue with the SET TRANSACTION command. In the case of a flagged "EXCEPTION" transaction the system administrator can take action by changing the state of the "EXCEPTION" transaction to that of "DONE" with the SET TRANSACTION/STATE=EXCEPTION /NEW_STATE=DONE to allow the recovery to continue.

4.1.2 Transaction State Changes

There are eight valid state changes allowed for the SET TRANSACTION command. Attempting to change transaction state to a state that is not allowed produces an error message of %RTR-E-INVSTATCHANGE, Invalid to change from current state to the specified state. The Table 6–19 table identifies the valid state changes.

Table 4–19 Valid Transaction State Transitions

	NEW STATE			
Current State	**COMMIT**	**ABORT**	**EXCEPTION**	**DONE**
SENDING		YES		
VOTED	YES	YES		
COMMIT			YES	YES
EXCEPTION	YES			YES
PRI_DONE				YES

Four typical situations are listed below where transaction state changes by the system administrator are allowed.

1. State SENDING changed to state ABORT.

   The application server, after receiving a rtr_mt_msg_1 message and before calling rtr_accept_tx( ) for a particular transaction, experiences a "hung" situation and cannot proceed. Aborting this transaction with the SET TRANSACTION command is the only way to correct it. Internally, RTRACP will send the ABORT message to the router as well as the all participating servers to abort this transaction in a consistent matter.

2. State VOTED changed to state COMMIT.

   This is the case where a application server running on the backend may have been separated from the rest of participating servers after casting the VOTE for the transaction. The other servers may have already committed the transaction but not "forgotten" it. As far as the application is concerned, this global transaction is committed and all changes have been committed to the underlying database on the different sites. However, the local transaction record is still in VOTED state in the RTR journal. You can use the command to manually commit the local transaction branch.

   Note that this command is only applicable if there is no coordinating router running, i.e., servers are separated from the rest of the RTR network. If this is not the case, RTR rejects the command.

3. State VOTED changed to state ABORT.

   In a similar manner to the VOTED-to-COMMIT situation described above, the server has been separated from the other participating servers and all other participants aborted this transaction; use this command to manually abort the local transaction branch.

   Note that this command is only applicable if there is no coordinating router running and servers are separated from the rest of the RTR network. If this is not the case, RTR rejects the command.

4. State COMMIT changed to state DONE.

This is the case where, for example, a server crashed while performing an SQL commit immediately after receiving a `mt_accepted` message. The transaction is in COMMIT state as recorded in the RTR journal and the transaction is also committed in the underlying database.

After the `SET TRANSACTION` command is executed the `DUMP JOURNAL` command can be used to verify the result.

5
RTR Monitoring

This chapter contains a description of the **RTR monitor**. The RTR monitor gives you a means of viewing the activities of RTR and your applications. Many different aspects of RTR's behaviour can be viewed, allowing the activities and performance of RTR to be analyzed.

5.1 Introduction

The RTR monitor provides a means to continuously display the status of RTR and the applications using it.

It can be used to check the correct operation of an RTR network, showing information useful for tuning, capacity planning, and locating configuration and application errors.

The information displayed is composed of named `data items` which are continuously updated by RTR. These data items can be displayed in various formats, and combined using simple arithmetic operators and constants.

The monitor is invoked with the `RTR MONITOR` command. RTR monitor displays a `monitor picture` that is periodically updated. See Section 6.2 for the full syntax of the `MONITOR` command.

A monitor picture contains elements that are either text (such as labels and titles) or variables derived from data items. Monitor pictures can be defined either interactively at the `RTR>` prompt or defined in a file called a `monitor file`.

You can use monitor files that are provided with RTR, and you can create your own. See Appendix A for information about creating monitor files.

5.2 Standard Monitor Pictures

A number of standard monitor pictures are supplied with RTR. These cover most of the usual monitoring requirements. You may define your own monitor pictures or alter the standard ones to suit your particular needs. Table 5–1 contains a list of the standard monitor pictures. To display one of these pictures use the following command at the RTR prompt:

```
RTR> MONITOR picture-name
```

The files for standard monitor pictures are installed on your system when RTR is installed. The location of these files is platform-specific. The filenames are the picture name appended with `.mon` (You type the filename without `.mon` when starting the display.)

Note

Obsolete monitor pictures have been removed from the documentation.

Table 5–1 Standard Monitor Pictures

Picture name	Description
accfail	Shows link transport name for links on which a connection attempt was declined, with a reason for failure. The most recent entry is highlighted.
acp2app	Displays counts of messages and number of bytes from RTRACP to the application, as viewed from a specific node.
active	Displays a list of RTR processes, and for each process the number of transactions they have started, the number of transactions they have completed and the number of transactions that are still active.
app2acp	Displays counts of messages and number of bytes from the application to RTRACP, as viewed from a specific node.
broadcast	Displays information about RTR user events by process, including number of user events enqueued, received, and discarded.
calls	Displays the total number of RTR API calls and their success or failure for the processes on all the nodes being monitored. All RTR message are also show by message type. (Pending messages are ones that an application has not received yet). Use the /IDENTIFICATION=process-id qualifier to display the values for one specific process, otherwise the total values for all processes are displayed.
channel	Displays the roles of the channels declared by an application. This can be useful as a debugging tool in the early stages of application development.
congest	Displays a sorted list of nodes responsible for causing the most congestion since RTR was last started, and the instantaneous state.
connects	Displays connection status summary, including the number of links up and down, and a list of links with state (up or down), architecture, network transport, and fail-reason, if any.
ddtm	Displays counts of RTR calls to DECdtm, as viewed from a specific node, for all PIDs, processes, and images.
dtx	Displays counts of RTR DTX calls including open, start, prepare, rollback, commit, and close, as viewed from a specific node for all PIDs, processes, and images.
dtxrec	Displays a summary of DECdtm transaction recovery (DTX), as viewed from a specific node for all PIDs, processes, and images.
event	Displays event routing data by facility. Information includes events in transit and destination information showing number of events enqueued, processed, and discarded.
facility	Displays a number of per facility data items. The /FACILITY qualifier can be used to say which facility should be monitored. If this is not specified then the totals of the data items for all facilities are displayed.
flow	Displays the flow control counters.
frontend	Displays frontend status and counts by node and facility, including frontend state current router, reject status, retry count, and quorum rejects.
group	Shows server and transaction concurrency on a partition basis.
ipc	Shows counts of inter-process communication (IPC) activity in the RTR ACP and active RTR applications.
ipcrate	Displays rate information on IPC messages, byte counts, and IO primitive usage.

(continued on next page)

Table 5–1 (Cont.) Standard Monitor Pictures

Picture name	Description
jcalls	Displays counts of successful (success), failed (fail) and total journal calls for local and remote journals.
journal	Displays the current journal usage on a node. Local node journal statistics are provided, and data for non-local journals accessed from the local node. Include statistics covering total number of entries and records written, the number of records read, and how many bytes were involved. Bar graphs showing current usage of journal blocks (as a percentage of the total) are also provided.
link	Displays a number of per link data items. The `/LINK=link-name` qualifier can be used if the values for one specific link are to be displayed, otherwise the total values for all links are displayed.
netbytes	Displays a list of the links to other nodes. For each link, the total number of bytes received and sent on that link and the number of bytes received and sent per second are displayed.
netstat	For each link, displays the connection status in detail, with the link state (up or down), and architecture type of remote node (such as VAX, I386, Alpha, and so on).
partit	Displays the status of server partitions. Shows the partition identifiers, key ranges and key segments, and the status of the servers (active, recovering and so on).
queues	Shows transaction queues on a partition basis.
quorum	Tracks (by facility) the configuration, reachability and quorum status of one or more nodes.
rdm	Displays memory used by each RTR subsytem.
recovery	Displays the status of server recovery procedures, such as waiting for quorum, catching up transactions, and so on.
rejects	Displays the last `rtr_mt_rejected` message received by each running process.
rejhist	Displays the last ten `rtr_mt_rejected` messages received by the selected process.
response	Displays the elapsed time that a transaction has been active on the opened channels of a process.
rfb	Displays router failback operations, including both a summary and detail by facility.
rolequor	A detailed picture of the various data items displayed in the QUORUM picture, separated by roles. If a quorum problem is encountered, this picture may be useful for problem diagnosis.
routers	Displays information on a router node. It gives an indication of the utilization of the router in terms of transactions and broadcasts routed through this node. Useful to monitor performance, or locate problems.
routing	Displays statistics of transaction and broadcast traffic by facility.
rscbe	Displays the most recent calls history for the RSC subsystem on a backend node.
rtr	Displays various per node data items.
stalls	Displays in real time any network links that are currently stalling in their outbound traffic, and provides a history of the stalls that the various links encountered during their lifetime.
system	Displays the state of critical resources within the RTR environment. If a resource has exceeded a predefined threshold, a warning indicator is displayed.
tps	Displays the rate of transaction commits carried out by each process using RTR.
tpslo	Displays low end of the rate of transaction commits carried out by each process using RTR.

(continued on next page)

Table 5–1 (Cont.) Standard Monitor Pictures

Picture name	Description
traffic	Displays a list of the links to other nodes. Shown for each link are: byte rate, packet rate, message rate and congestion, in both directions. Average packets per second is also shown.
trans	Displays transactions for a frontend, router and backend.
v2calls	Shows RTR Version 2 verb usage through the interoperability subsystem. The screen layout is identical to the RTR Version 2 monitor calls picture.
xa	Displays XA counter information including success and failure as well as call and readonly counters.

The following sections describe the more commonly used standard monitor pictures in detail.

5.2.1 Monitor ACCFAIL (Link Acceptance Failures)

When configuring RTR it can happen that nodes sometimes fail to connect up. Whilst the cause of the error can be viewed on the initiator side with the MONITOR NETSTAT picture, it can be difficult to pinpoint the problem when looking at the other end of the link. The monitor picture ACCFAIL can be used to display the reason for the local node to refuse to accept connections. An example:

```
================================================================================
                    U n a c c e p t a b l e    L i n k s

         Most recent links on which a connection attempt was declined

            Node: LENGTH                   Wed Jan  7 1998 10:51:00

--------------------------------------------------------------------------------
Link Transport Name(s)                  Reason for failure
--------------------------------------------------------------------------------
ilira                                   node is not configured for the facility
dmark.zko.dec.com                       node not recognized
breal                                   facility name not matched
DMARK DEC:.ZKO.DMARK                    node not recognized
ilira                                   node is not configured for the facility
dmark.zko.dec.com                       node not recognized
breal                                   facility name not matched
sfranc                                  node role definitions do not match for
breal                                   facility name not matched
DMARK DEC:.ZKO.DMARK                    node not recognized
--------------------------------------------------------------------------------

         List entries are reused in a cyclical fashion; the most
         recent entry is highlighted.

================================================================================
```

Some of the errors that can be displayed by ACCFAIL are:-

- RTR_STS_NOTRECOGNISED - "node not recognized"

  The local node has received a connection request over a link that is not configured in any facility on the local machine. If you expected the connection to succeed as the result of having a template link defined, check carefully the names displayed in the monitor picture. These are the names obtained from the network transport over which the link connection attempt was received, and it is with these that a match with a template (wildcard specification) link must succeed. Bear in mind that the name match is performed in a case sensitive manner. Names corresponding to TCP links may or may nor contain

your domain name, depending on how the name is entered in either your local hosts file or name server. DECnet-Plus systems may yield both a pseudonym and a link name; both are checked for a match with a template.

- RTR_STS_FACNOTDEC - "facility name not matched"

  The connecting link is configured, but the facility that it requests does not exist on the local node.

- RTR_STS_NODENOTCNFG - "node is not configured for the facility"

  The connecting link is configured on the local node, but not as part of the requested facility.

- RTR_STS_ROLESMISMATCH - "node role definitions do not match for this facility"

  The connecting link is configured in the requested facility, but with a different role configuration on the local node.

- RTR_STS_TRNOQUO - "router has no quorum in facility %s"

  The router is unable to accept front end connections since it is not quorate. This condition will clear up automatically as soon as the router gains connections to sufficient backends.

5.2.2 Monitor ACP2APP

```
RTR ACP to Application Messages, Node: NodeA  PID: -ALL-   Process name: -ALL-
    Image: -ALL-                                        14:15:46 Mon Jan 25 1999
messages            client               server             other          pend
                    #      Bytes         #     Bytes        #      Bytes     #
opened              0          0         0         0        0          0     0
closed              0          0         0         0        0          0     0
msg1                                      0         0                       0
msg1_uncertain                            0         0                       0
msgn                                      0         0                       0
repl_2_client       0          0                                            0
rettosend           0          0                                            0
accepted            0          0         0         0                       0
rejected            0          0         0         0                       0
user_event          0          0         0         0                       0
rtr_event           0          0         0         0                       0
mt_prepare                                                  0          0     0
                    other
request_info                                                0          0     0
set_info                                                    0          0     0
                  calls   active     fail   timeout
receive_message       0        0        0         0
user_wakeup           0        0
```

Displays counts of messages and number of bytes from RTRACP to the application, as viewed from a specific node. Includes openend, closed, msg1, msg1_uncertain, msgn, repl_2_client (reply to client), rettosend (return to sender), accepted, rejected, user_event, rtr_event, mt_prepare, request_info, and set_info messages as appropriate. For receive_message and user_wakeup, displays calls, active, fail, and timeout counts.

The default is to display information on all PIDs, process names, and images. To display information on one process only, user the qualifer /IDENTIFICATION=process-id.

5.2.3 Monitor Active

```
ACTIVE TRANSACTIONS BY PROCESS  Fri Mar 12 1999 19:32:41

                                                 Starts    Completions    Active
All processes:                                        5              5         0

Node      ID      Process        Image
NodeA     11141   11141          rtr                 5              5         0
```

Displays a list of RTR processes, and for each process the number of transactions they have started, the number of transactions they have completed and the number of transactions still active.

5.2.4 Monitor APP2ACP

```
RTR Application to ACP Messages, Node: NodeA  PID: -ALL-   Process name: -ALL-
    Image: -ALL-                                          14:21:19 Mon Jan 25 1999
messages             client                 server                 other
                     #      Bytes           #     Bytes            #      Bytes
open_channel         0          0           0         0            0          0
close_channel        0          0           0         0            0          0
accept_tx            0          0           0         0
reject_tx            0          0           0         0
broadcast_event      0          0
start_tx             0          0
send_to_server       0          0
reply_to_client                             0          0
request_info                                                      0          0
set_info                                                          0          0
```

Displays counts of messages and number of bytes from the application to RTRACP, as viewed from a specific node. Includes open_channel, close_channel, accept_tx (accept transaction), reject_tx (reject transaction), broadcast_event, start_tx, send_to_server, reply_to_client, request_info, and set_info.

The default is to display counts for all PIDs and processes, for client, server, and other roles.

5.2.5 Monitor Broadcast

```
                BROADCAST RECEPTION BY PROCESS 15:20:27   6-APR-1999

Node      ID    Process        Received Queued  Lost    Rate of delivery
Total                              2750     5     17               850.0

NODEA  20400249 RTRACP                0     0      0
NODEA  2040024D BATCH_2993         2750     5     17               850.0
NODEA  2040024B BATCH_2991            0     0      0
NODEA  2040024C BATCH_2992            0     0      0
```

Displays information about the RTR user events process. Fields displayed included number of user events enqueued for the application, number of user events received by the application, and number of user events discarded by RTR.

5.2.6 Monitor Calls

```
RTR api calls, Node: nodea.zuo.dec.com  , PID: 2162    , Process name: -ALL-
Image: -ALL-                                          Fri Feb 12 1999  16:38:05
CALLS            client   server  fail  MESSAGES           client   server  pend
open_channel          1        1     0  mt_opened               1        1     0
close_channel         0        0     0  mt_closed               0        0     0
start_tx              0              0  mt_msg1                          0     1
send_to_server        1              0  mt_msg1_uncertain                0     0
                                        mt_msgn                          0     0
reply_to_client               0     0  mt_reply                0              0
                                        mt_rettosend            0              0
prepare_tx            0              0  mt_prepared             0              0
accept_tx             0        0     0  mt_accepted             0        0     0
reject_tx             0        0     0  mt_rejected             0        0     0
broadcast_event       0        0     0  mt_user_event           0        0     0
set_user_handle       0        0     0  mt_rtr_event            0        0     0
get_tid               0        0     0  mt_prepare                       0     0
                 other                                    other
request_info          3              0  mt_request_info         2              0
set_info              0              0  mt_set_info             0              0
error_text            2                 mt_closed               2
set_wakeup            0
                 calls   active     fail  timeout
receive_message       9        1        2        2
user wakeup           0        0
```

Displays the total number of RTR API calls and their outcome for the processes on all the nodes being monitored. Use the /IDENTIFICATION=process-id qualifier to display the values for one specific process, otherwise the total values for all processes are displayed.

5.2.7 Monitor Channel

```
RTR CHANNELS BY TYPE PER PROCESS  Fri Feb 12 1999 16:41:13

                                          Client    Server       Call-out
Node    ID      Process        Image             Pri. Sby.  Router  Backend
nodea  2162    2162                        1        0    0       0        0
```

Displays the channels opened by RTR CALL RTR_OPEN_CHANNEL comands.

5.2.8 Monitor Connects

```
              C o n n e c t i o n   S t a t u s   S u m m a r y

           Node: nodea.zuo.dec.com         Tue Feb 16 1999 13:02:18

         -Executive summary---------------------------------------
               Number of links up:     3        (100.%)
             Number of links down:     0        (0.0 %)
         ------------------------------------------------------

-Detail-------------------------------------------------------------------------
 Node -> Link  State  Arch T'port Fail-reason
-------------:-----:-----:------:-----------------------------------------------
nodea->nodea      up alpha      -
nodea->nodeb      up alpha    TCP
nodea->nodec      up  i386    TCP
```

Displays the link protocol for connected links, and the fail reason as a text message for any links on which a connection has failed. Unconnected links where connection have been attempted are highlighted. Link state and architecture of the remote node are also displayed. Summarizes link status and is less detailed than the monitor netstat display.

5.2.9 Monitor Event

```
       EVENT ROUTING STATISTICS BY FACILITY  6-APR-1999 15:21:47

                                          Destination          Transit
Node    Facility                    In     Out  Lost    In    Out    Lost
Total                              180     175    5   180    180       0

NODEA   FACCMS                      25      25    0    25     25       0
NODEA   TESTFAC                    155     150    5   155    155       0
```

Displays event routing data by facility. Information includes events in transit from RTR to a destination facility and destination information showing number of events enqueued for the application (In column), number of events processed by the application (Out column), and the number of events discarded by RTR (Lost column).

5.2.10 Monitor Facility

```
 FACILITY COUNTERS  7-JAN-1999 14:04:27, NODE: -ALL- , FACILITY: -ALL-

ASM_MSGS_TO_APPS  290005  TMBE_TX_RQ_COMMITS        0  NCF_TR_FE_LOSS         0
ASM_MSGS_FROM_APPS290404  TMBE_TX_RQ_ABORTS         0  NCF_TR_BE_LOSS         0
BM_NCF_EVENTS_DELV     6  TMBE_TX_ACCEPTS       72501  NCF_TR_FE_GAIN         1
BM_NCF_EVENTS_RCVD     7  TMBE_TX_REJECTS           0  NCF_TR_BE_GAIN         1
TM_NCF_EVENTS_RCVD    13  TMBE_TX_RTR_FORGETS   72499  NCF_TR_FQM_LOSS        0
TMFE_TX_RQ_STARTS  72700  TMBE_CRPS_REQUESTED       0  NCF_FE_TR_LOSS         0
TMFE_TX_RQ_ENQS    72700  RSC_GETTXDST_CALLS    72700  NCF_FE_TR_NOCUR        0
TMFE_TX_RQ_COMMITS 72700  RSC_GETTXDST_SUCCESS  72700  NCF_FE_TR_GAIN         1
TMFE_TX_RQ_ABORTS      0  RSC_GETTXSRV_CALLS        0  NCF_BE_TR_LOSS         0
TMFE_TX_ACCEPTS    72500  RSC_GETTXSRV_SUCCESS      0  NCF_BE_TR_GAIN         1
TMFE_TX_REJECTS        0  RSC_GETEVTDST_CALLS       0  NCF_FQM_TR_LOSS        0
TMFE_TX_REPLAYS        0  RSC_GETEVTDST_SUCCESS     0  QRM_REQ_LINK_QUEUE     0
TMRT_TX_RQ_STARTS  72700  RSC_GETEVTRCV_CALLS       6  QRM_RSP_LINK_QUEUE     0
TMRT_TX_RQ_ENQS    72700  RSC_GETEVTRCV_SUCCESS     0
TMRT_TX_RQ_COMMITS 72700  RSC_FE_NODES              1
TMRT_TX_RQ_ABORTS      0  RSC_BE_NODES              1
TMRT_TX_ACCEPTS    72500  RSC_TR_SERVER_CLASSES     1
TMRT_TX_REJECTS        0  RSC_BE_SERVER_CLASSES     1
TMBE_TX_RQ_STARTS  72700  RSC_SERVER_CHANS          1
TMBE_TX_RQ_ENQS    72700  RSC_REQUESTER_CHANS       2
```

Displays per facility counters. Use the /FACILITY qualifier to specify a facility; if it is not specified then the totals of the counters for all facilities are displayed.

5.2.11 Monitor Flow

```
 FLOW CONTROL COUNTERS  7-JAN-1999 14:08:06, NODE: -ALL- , FACILITY: -ALL-

          CREDIT      DATA RATE              REQUESTS                    GRANTS
  ROLE   AVAILABLE    BYTES/SEC       WAITS   SENT  PENDING       SENT  PENDING
FE=>TR       15000         2065         307      966     0          966     0
TR=>BE       15000         2065          70      998     0          998     0
BE=>TR           0            0           0        0     0            0     0
TR=>FE       15000            0           0        2     0            2     0

     LINK     DATA RATE      WAITS     PENDING   REQS SENT      CACHE IN USE
NODEA =>NODEA          0          0           0           1      NODEA          0
NODEA =>NODEB          0          0           0           0      NODEB      51456
NODEB =>NODEB       2065        307           0         968
NODEB =>NODEA       2065         70           0         999
```

Displays the flow control internals.

5.2.12 Monitor Group

```
% rtr monitor group

  Concurrency Measures    Tue Apr 6 1999 10:04:26, NODE: NODEA

                                                                    --  averages --
                         txn  -server-  --  transactions --        srv   txn   txn
krid       state         cnt  cnt  act  vreq vote  ack /csn        act  /sec  /csn
16777216  shd_rec_fail     0    1     0    0    0    0    0        0.0   0.0   0.0
16777217  shd_rec_fail     0    1     0    0    0    0    0        0.0   0.0   0.0
```

Table 5–2 MONITOR GROUP Fields

Field	Meaning
krid	Key range (partition) identifier.
state	Partition state.
txn cnt	Number of transactions executed for this partition.
srv cnt	Number of servers active for this partition.
srv act	Number servers that are currently busy processing txns for this sample.
The following fields track the progress of a transaction through the states: vote requested, voted, acknowledged.	
vreq	Number of transactions that are waiting for the server to vote.
vote	Number of transactions that have been voted on by the server but not committed by RTR.
ack	Number of transactions that have been committed but have not been acknowledged by the server. Acknowledgment occurs on a subsequent `rtr_receive_message( )` call by the server processing this transaction to get a message for a new transaction.
/csn	Number of transactions which have been grouped under the same "commit sequence number" (CSN). This grouping determines the ordering of transactions submitted to a secondary shadow server.
txn/sec	The average rate of transaction starts per second for this partition.
txn/csn	average number of transactions which have been grouped under the same commit sequence number (CSN) since this partition became active. This average is computed as the quotient of the `txn cnt` column and the total number of CSN's.

5.2.13 Monitor IPC

```
RTR> Monitor IPC
Node: LENGTH                 I P C   S u m m a r y    Fri Mar  5 1999 11:18:34

 +------------------------------------------------------------------------------+
  This screen displays usage information on IPC messages, byte counts and IO
  primitives. Display units are counts, kbytes and calls respectively.
 +------------------------------------------------------------------------------+

         | - - - - - O u t g o i n g / s - - - - | - - I n c o m i n g / s - -|
Process    Messages ...kbytes    send() ...kbytes  Messages    recv() ...kbytes
rtracp       110334     49744     73437     49744     73434    220299      5616
3123B84F          0         0         0         0         0         0         0
31232395      73282      5569     73280      5569    109930    293144     49685
```

Displays interprocess communication message information.

5.2.14 Monitor IPCRATE

```
RTR> Monitor IPCRATE
Node: LENGTH                 I P C    R a t e s      Fri Mar  5 1999 11:18:53

 +------------------------------------------------------------------------------+
  This screen displays rate information on IPC messages, byte counts and IO
  primitive usage. Display units are counts, kbytes and calls per second
  respectively.
 +------------------------------------------------------------------------------+

         | - - - - - O u t g o i n g / s - - - - | - - I n c o m i n g / s - -|
Process    Messages ...kbytes    send() ...kbytes  Messages    recv() ...kbytes
rtracp           44        19        29        19        29        86         2
3123B84F          0         0         0         0         0         0         0
31232395         28         2        28         2        41       110        18
```

Displays interprocess communication rate information for messages.

5.2.15 Monitor Journal

```
  JOURNAL USAGE ON NODE NODEA           AT 10:36:05 Tue Apr  6 1999

  LOCAL JOURNAL                          STANDBY JOURNAL(S)

  JNL_LCL_BLOCKS_IN_USE       128        JNL_RMT_BLOCKS_IN_USE          0
  [___            13%          ]        [               0%            ]
  JNL_LCL_NR_BLOCKS           992        JNL_RMT_NR_BLOCKS            992
  JNL_LCL_TOP_BLOCKS_USED     128        JNL_RMT_TOP_BLOCKS_USED        0
  JNL_LCL_BLOCKS_AVAILABLE    864        JNL_RMT_BLOCKS_AVAILABLE       0
  JNL_LCL_TX_ENTRIES            1        JNL_RMT_TX_ENTRIES             0
  JNL_LCL_TX_RECORDS            2        JNL_RMT_TX_RECORDS             0
  JNL_LCL_MEMORY_BYTES     530121        JNL_RMT_MEMORY_BYTES        4197
  JNL_LCL_DISK_READS           31        JNL_RMT_DISK_READS            33
  JNL_LCL_BLOCKS_READ         992        JNL_RMT_BLOCKS_READ         1056
  JNL_LCL_DISK_WRITES          12        JNL_RMT_DISK_WRITES            0
  JNL_LCL_BLOCKS_WRITTEN       14        JNL_RMT_BLOCKS_WRITTEN         0
  JNL_LCL_ENTRIES_TOTAL       201        JNL_RMT_ENTRIES_TOTAL          5
  JNL_LCL_RECORDS_TOTAL       398        JNL_RMT_RECORDS_TOTAL          9
  JNL_LCL_RECORDS_READ         21        JNL_RMT_RECORDS_READ           0
  JNL_LCL_REC_BYTES_READ     8006        JNL_RMT_REC_BYTES_READ         0
  JNL_LCL_NONTX_ENTRIES         5        JNL_RMT_NONTX_ENTRIES          0

  JNL_LCL_OPEN_JOURNALS    2
```

Displays information about journal usage, including total number of entries and records written, number of records read, and how many bytes were involved. Bar graphs showing current usage of journal blocks (as a percentage of the total) are also provided.

The local journal figures refer to journal usage for the displayed node. Standby journals are journals of standby nodes that are being accessed due to restart or catch-up situations. Under normal conditions standby journal figures are all zero.

The bar graphs appear under the first line of the display (JNL_LCL_BLOCKS_IN_USE).

5.2.16 Monitor Link

```
    LINK COUNTERS  8-MAR-1999 14:24:44, NODE: -ALL- , LINK: -ALL-  -> -ALL-

NCF_PROTOCOL             0        NIO_WRITING              0
NCF_TIMEOUT              0        NIO_READING              0
NCF_LINKEXIT             0        NIO_WRERROR              0
NCF_DISCONNECT           0        NIO_RDERROR              0
NCF_THIRDPARTY           0        NIO_RDINCMP              0
NCF_PATHLOST             0        NIO_SEQERR               0
NCF_RESPONSES_SENT       3        NIO_BUFOVF               0
NCF_QUERIES_RCVD        11        NIO_READS_ACTIVE         3
NCF_RESPONSES_RCVD       2        NIO_WRITES_ACTIVE        0
NCF_QUERIES_SENT        10        NIO_BYTES_RCVD    44206961
NCF_IPT_MSGS_RCVD       16        NIO_BYTES_SENT    44206996
NCF_IPT_MSGS_SENT       16        NIO_PCKTS_RCVD      412114
NCF_LINK_GAIN            4        NIO_PCKTS_SENT      412114
NCF_LINK_LOSS            0        NIO_MSGS_RCVD       189284
NCF_ABORTED              0        NIO_TMO_SENDS       161307
NCF_REJECTEE             1        QRM_REQ_LINK_QUEUE       0
NCF_ACCEPTED             1        QRM_RSP_LINK_QUEUE       0
NCF_CONFIRMED            1
NCF_INITIATED            2
```

Displays a number of per link counters. The /LINK=link-name qualifier can be used if the values for one specific link are to be displayed, otherwise the total values for all links are displayed.

5.2.17 Monitor Netbytes

```
LINK TRAFFIC IN BYTES Fri Apr 16 1999 17:41:12, NODE: nodea.zko.dec.com

                          Bytes  Rcvd                 Bytes Sent
                      -------------------------    -------------------------
                           Abs      Rate     Max          Abs     Rate     Max
Total                 59201466    6776.0      -    1002579480 113857.0      -

nodea->nodea            3072248     336.0   336.0      3072248    336.0    336.0
nodea->nodeb           42569496    4974.0  4974.0    717457678  84438.0  84438.0
nodea->nodec           13559722    1466.0  1466.0    282049554  29083.0  29083.0
```

Displays a list of the links to other nodes. For each link, the total number of bytes received and sent on that link and the number of bytes received and sent per second are displayed. Derived from the NIO_BYTES_RCVD and NIO_BYTES_SENT counters. The Max field represents the maximum rate since the link started.

5.2.18 Monitor Netstat

```
                C o n n e c t i o n   S t a t u s   D e t a i l
                Node: NODEA                    Mon March 15 1999 09:50:28
                  Ini  Cnf  Acc  Abo  Rej Loss Gain Ctmo Rstr State  Type FailCode
Node    Link       12    0    2   12   12    1    3    0    0
NODEA ->nodeb       0    0    0    0    0    0    1    0    0    up alpha
NODEA ->nodec       6    0    0    6    6    0    0    0    0  down     ? 76490676
NODEA ->noded       6    0    0    6    6    0    0    0    0  down     ? 76490676
NODEA ->nodee       0    0    2    0    0    1    2    0    0    up alpha
```

Displays the link status for connected links in detail and the fail code for any links on which a connection has failed. Unconnected links where connection have been lost are highlighted. Link aborts, rejects, loss, gain, restarts, state and architecture of the remote node are also displayed. More detail included than in in the monitor connects display.

5.2.19 Monitor Partit

```
PARTITION DEFINITIONS  Tue Apr  6 1999 10:40:31, NODE: NODEA
Partition-name                              #    #      bounds      callout
                                 State     svrs segs   lo      hi    type
RTR$DEFAULT_PARTITION_16777218   active       1    1  "A"   "A"     -
RTR$DEFAULT_PARTITION_16777221   active       1    1  "B"   "B"     -
RTR$DEFAULT_PARTITION_16777217   active       1    1  0     429496
```

Partitions are shown in the form node$facility_partition-id or partition name if a partition name has been specified using the SET PARTITION command. The number of servers and key segments are shown for each partition. The least significant byte of the partition's low and high bound is also shown, and callout type (if any). The partition state meanings are given in Table 5–3.

Table 5–3 Monitor Partition States

State	Meaning
wt_tr_ok	Server is waiting for routers to accept it
wt_quorum	Server is waiting for backend to be quorate
lcl_rec	Local recovery
lcl_rec_fail	Primary server waiting for access to a restart journal
lcl_rec_icpl	Getting next journal to recover from
lcl_rec_cpl	Processed all journals for local recovery
shd_rec	Shadow recovery
shd_rec_fail	Shadow server waiting for access to a restart journal
shd_rec_icpl	Shadow getting next journal to recover from
shd_rec_cpl	Processed all journals for shadow recovery
catchup	Secondary is catching up with primary
standby	Server is declared as standby

(continued on next page)

Table 5–3 (Cont.) Monitor Partition States

State	Meaning
active	Server is active
pri_act	Server is active as primary shadow
sec_act	Server is active as secondary shadow
remember	Primary is running without shadow secondary

5.2.20 Monitor Queues

```
        TRANSACTION QUEUES BY PARTITION 15-JAN-1999 12:42:53, NODE: NODEA

Partition-name                              Processed          Queued     #
                                          Txns    Msgs  Rplys    Txn  Msg   Svrs
NODEA$NODEB$16842753                      5792    5794      0      2    6    3
```

Shows transaction queues on a partition basis. Uses counters from Transaction Manager (TM) and the Requester/Server configurator (RSC).

5.2.21 Monitor Quorum

```
QUORUM STATUS BY NODE AND FACILITY Tue Apr  6 1999 10:50:24, NODE: NODEA
(node/role counts can be inaccurate for incorrectly configured facilities)
States: bad configuration,not connected,minority,uncertain,quorate     node/roles
Node     Facility                       State                          CNF RCH QRT
NODEA    RTR$DEFAULT_FACILITY           TR:quorate,BE:quorate            2   2   2
NODEA    shadow                         TR:quorate,BE:minority(ffranc)   4   2   1
```

Quorum states are shown for router (TR) and backend (BE) nodes and roles in the columns State.

The number of nodes seen as configured (CNF), reachable (RCH) and quorate (QRT) are shown for each node, in the columns node/roles.

5.2.22 Monitor Recovery

```
RECOVERY INFORMATION at Tue Apr  6 1999 10:54:50 on NODEA

                          Last        Restart-Recovery     Shadow-Recovery
              Server      Recovery    Journal  Txns        Journal  Txns
Partition-id  State       Backend     Scans    Recovered   Scans    Recovered
------------  ------      -------     -------  ---------   -------  ---------
16777218      active      NODEA             1          0         0          0
16777221      active      NODEA             1          0         0          0
16777217      active      NODEA             1          0         0          0
```

Shows the progress of transaction recovery. Last recovery backend is the last backend accessed to recover transactions. If the server state is lcl_rec_fail or shd_ rec_fail, this entry is the name of the background which could not be accessed. Journal scans is the number of journal files searched. Transactions recovered is the number of transactions found for this partition.

Server recovery state meanings are shown in Table 5–4.

Table 5–4 Monitor Recovery States

State	Meaning
wt_tr_ok	Server is waiting for routers to accept it
wt_quorum	Server is waiting for backend to be quorate
lcl_rec	Local recovery
lcl_rec_fail	Primary server waiting for access to a restart journal
lcl_rec_icpl	Getting next journal to recover from
lcl_rec_cpl	Processed all journals for local recovery
shd_rec	Shadow recovery
shd_rec_fail	Shadow server waiting for access to a restart journal
shd_rec_icpl	Shadow getting next journal to recover from
shd_rec_cpl	Processed all journals for shadow recovery
catchup	Secondary is catching up with primary
standby	Server is declared as standby
active	Server is active
pri_act	Server is active as primary shadow
sec_act	Server is active as secondary shadow
remember	Primary is running without shadow secondary

5.2.23 Monitor Rejects

```
                          Rejected Transaction Summary
NODE: NODEA                    PROCESS: 20413894      Fri Apr  9 1999 10:26:14

        Time           Pid     Chan    Reason   Status Text
------------------   ------  ------  --------  ----------------------------
Fri Apr  9 10:18:43  20417266 client         0  No server available to handle
Fri Apr  9 10:17:47  20417274 server         0  Client aborted tx
```

Displays the last `rtr_mt_rejected` message received by each running process.

Table 5–5 MONITOR REJECTS Fields

Field	Meaning
Time	Time of day that the `rtr_mt_rejected` message was received.
Pid	The process id that received the message.
Chan	The type of channel (client or server) that received the message.
Reason	The reason field returned in the `rtr_status_data_t` buffer.
Status Text	The textual status that describes the reject reason.

5.2.24 Monitor Rejhist

```
                              Rejected Transaction History
NODE: NODEA                        PROCESS: 38009A8B      Mon Mar  9 1999 10:26:14

        Time           Chan    Reason   Status Text
------------------   ------   -------  -----------------------------------
Mon Mar 15 18:06:06   client         0  Client aborted tx
Mon Mar 15 18:06:41   server         0  Normal successful completion
Mon Mar 15 18:06:41   client         0  Server aborted tx

------------------------------------------------------------------------------

                        number of reject msgs = 3
                        number of accept msgs = 0
                        rejected / total txns = 100%
```

Displays the last ten rtr_mt_rejected messages received by the selected process. This picture should always be invoked with the /ID qualifier. The transaction identifier associated with the rejected transaction can be displayed with the SHOW PROCESS <id>/COUNTER=api_reject* command.

Table 5–6 MONITOR REJHIST Fields

Field	Meaning
Time	Time of day that the rtr_mt_rejected message was generated.
Chan	The type of channel (client or server) that received the message.
Reason	The reason field returned in the rtr_status_data_t buffer.
Status Text	The textual status that describes the reject reason.

5.2.25 Monitor Response

```
                    TRANSACTION DURATION AT 10:24:51 Fri Apr  9 1999

 Process     Image          Client Response Time        Server Response Time
   ID        Name         seconds 0   1   2   3   4   seconds 0   1   2   3   4
20413894   SERVER.EXE;4    0.000                       3.670
20417266   RTR.EXE;75      2.200                       3.440
20417274   SERVER.EXE;4    0.000                       1.160
```

Displays the elapsed time that a transaction has been active on the opened channels of a process. On the client, transaction duration is measured between the rtr_start_tx or rtr_send_tx call and the receipt of the final rtr_mt_accepted or rtr_mt_rejected message. A call to rtr_reject_tx also marks the end of a transaction. On the server, transaction duration is measured between receipt of a rtr_mt_msg1 or rtr_mt_msg1_uncertain message and the receipt of the final rtr_mt_rejected message or rtr_reject_tx call. Accepted transaction end times are recorded when the server issues a rtr_receive_message call to request a new transaction for processing.

5.2.26 Monitor Rolequorum

```
  QUORUM COUNTS BY FACILITY  7-JAN-1999 14:32:48, NODE: -ALL-

     Router View of           Backend View of
         backends    routers       backend      routers
       CNF RCH QRT  CNF RCH QRT  CNF RCH QRT  CNF RCH QRT

VIP                          1   1   1    1   1   1    1   1   1    1   1   1
```

5.2.27 Monitor Routers

```
              ROUTER TRANSACTION COUNTERS AT 14:33:29  7-JAN-1999

Node:     -ALL-
Facility: -ALL-

            Abs        Rate         10  20  30  40  50  60  70  80  90 100
Starts:      116641    25.7
Enqueues:    116641    25.7
Commits:     116641    25.6
Aborts:           0     0.0
```

Displays information on a router node. It gives an indication of the utilization of the router in terms of transactions and broadcasts routed through this node. Useful to monitor performance, or locate problems. Uses counters in the Transaction Manager (TM) subsystem.

5.2.28 Monitor Routing

```
ROUTING STATISTICS BY FACILITY  Thu Apr-15-1999 14:34:20, NODE: -ALL-

                                      Transactions         Broadcasts
                                    Absolute    Rate   Absolute    Rate
Total                                 118489    39.2      68444   994.0

VIP                                   118489    39.2      68444   994.0
```

Displays statistics of transaction and broadcast traffic by facility. Rate is the number of transactions or broadcasts per second within the monitoring interval.

5.2.29 Monitor RSCBE

```
RTR> Monitor rscbe

Most Recent RSC Dclsrv Calls History on Backend LENGTH Thu Mar  4 1999,15:19:41

Key Range Id:   16777216     Partition Start Time: THU MAR  4 15:18:22 1999
Image Name:   RTR.EXE

T-delta  RSC calls          router           state                seq_nr
       0 send_dcl_to_master  sfranc           wait_for_response          0
       1 recv_status_ok      sfranc           rstart_rvy                 1
       1 send_dcl_to_master  sfranc           rstart_rvy_incomp          1
       1 recv_status_ok      sfranc           rstart_rvy                 1
       1 send_dcl_to_master  sfranc           rstart_rvy_incomp          1
       1 recv_status_ok      sfranc           active                     1
       1 send_dcl_to_other   depth            active                     1
       1 recv_status_ok      depth            active                     1
```

Displays backend request to server messages and declared state for routers

5.2.30 Monitor RTR

```
RTR> Monitor RTR

                  RTR COUNTERS  7-JAN-1999 14:35:05, NODE: -ALL-

ACP_WAKEUPS         310484   QRM_QCXTS              48
ACP_WAKE_REQS       859200   QRM_QIRS                0
CM_BYTES_PRESENT   1048576   QRM_RAES                0
CM_BYTES_IN_USE      51968   QRM_MAXQUOTA       278718
CM_FREECHCPCKT        3738   QRM_CURQUOTA       278718
TIM_TIMER_SETS      238349   QRM_RDES               18
TIM_TIMER_CANCELS    34165   QRM_QARS                0
TIM_TIMER_DELIVERS  204174   QRM_QUERIES_SENT   481308
TM_QRM_QUERIES_SENT      0   QRM_QUERIES_RCVD   481308
TM_QRM_QUERIES_RCVD      0   QRM_RESPONS_SENT     3082
NCF_REJECTEE             1   QRM_RESPONS_RCVD     3092
NCF_REJECTER             1   QRM_RETRIES             7
NCF_FACILITY_UP          2   QRM_TIMEOUTS            0
NCF_FACILITY_DOWN        0
RSC_ALLOC_MEM         2098
```

Displays various per node counters.

5.2.31 Monitor Stalls

```
                NETWORK STALLS AT 29-JAN-1999 15:35:03, ON NODE: TR1
                     QIOs      Bytes  Link             Stalls
                  Issued Rate    Sent Drops Secs  <3s  <10s  <30s  >30s Tot
Total               5467  0.0  327148   2     33   23     1     0     0  24
TR1 -> TR1            29  0.0    3718   0      0    0     0     0     0   0
TR1 -> FE2           509  0.0   20707   0      4    4     0     0     0   4
TR1 -> BE1           303  0.0   13707   0      3    3     0     0     0   3
TR2 -> TR2           111  0.0   11682   0      0    0     0     0     0   0
TR2 -> BE1           504  0.0   22743   0     18    8     1     0     0   9 Stall
FE1 -> FE1            64  0.0    6645   0      0    0     0     0     0   0
FE1 -> FE2           373  0.0   18890   0      2    2     0     0     0   2
FE1 -> BE1           310  0.0   24487   0      0    0     0     0     0   0
FE2 -> FE2           231  0.0   18900   0      0    0     0     0     0   0
FE2 -> BE1           284  0.0   22503   1      1    1     0     0     0   1
FE2 -> TR1           536  0.0   28166   0      0    0     0     0     0   0
FE2 -> FE1           396  0.0   23643   0      0    0     0     0     0   0
BE1 -> BE1           355  0.0   28121   0      0    0     0     0     0   0
BE1 -> FE2           284  0.0   13014   1      0    0     0     0     0   0
BE1 -> TR2           515  0.0   27502   0      2    2     0     0     0   2
BE1 -> TR1           328  0.0   25698   0      1    1     0     0     0   1
BE1 -> FE1           335  0.0   17022   0      2    2     0     0     0   2
```

Displays in real time any network links that are currently stalling (that is, waiting to transmit outbound traffic) and provides a history of the stalls that the various links have encountered during their lifetime. The display shows:

The total number of network QIOs issued,
Rate of network QIOs per second,
Total number of bytes sent,
Number of DECnet link losses,
Total waited seconds for QIO completions,
Number of 1 to 2 second stalls,
Number of 3 to 9 second stalls,
Number of 10 to 29 second stalls,
Number of stalls longer than 30 seconds,
Total number of stalled QIOs,
Current number of incomplete network writes.

5.2.32 Monitor System

```
                System Status at 10:27:51 Fri Apr  9 1999
                        node: NODEA

     Resource                OK   Warning
Facility QUORUM states......           x

JOURNAL free space.........  x

                                              Note: Additional detail
Link CONNECTS..............            x      about a resource can be
                                              obtained by monitoring
Link traffic STALLS.........  x               the subsystem specified
                                              in capital letters.
FLOW control credits........  x
                                              For example, to get more
PARTITION states............  x               information on links
                                              type MONITOR CONNECTS
CALL Msg outstanding..........           x
                                              To modify threshold values,
Transaction QUEUES............ x             edit the file SYSTEM.MON.

Transaction REJECTS........... x

Broadcast EVENT discards...... x
```

Displays the state of critical resources within the RTR environment. If a resource has exceeded a predefined threshold, a warning indicator is displayed.

The default thresholds are as follows:

Quorum	Warn if any roles are inquorate.
Journal	Warn if journal free space is less than 30% of total.
Links	Warn if any link is disconnected.
Stalls	Warn if 10 second stalls are greater than 1% of all messages sent.
Flow	Warn if the wait is more than one second for 10% of the total credit requests.
Partition	Warn if any of the partitions are not in one of the following states: Standby, Active, Pri_act, or Sec_act.
Calls	Warn if any messages have been pending for more than 30 seconds.
Queues	Warn if the transaction queue cannot be emptied within 10 seconds.
Rejects	Warn if the number of rejects (non-user) is greater than 5% of the total transactions processed, or a reject (non-user) has occurred within the last 30 minutes.
Events	Warn if the number of discards is greater than 5% of the total events sent.

Threshold values can be customized by editing the file SYSTEM.MON.

5.2.33 Monitor TPS

```
                TRANSACTION COMMITS BY PROCESS 14:37:23  7-JAN-1999

                                      Commits
Node      ID      Process           Abs.  Rate    10 20 30 40 50 60 70 80 90 100
-ALL-  00000000  -REQUESTERS-    123298  31.8
-ALL-  00000000  -SERVERS-      123297  28.9
NODEA  20200BEA  RTRACP             0   0.0
NODEA  20200C03  ANDERS_1       123297  28.9
NODEB  21400724  RTRACP             0   0.0
NODEB  214005F0  ANDERS_1       123298  31.8
```

Same as TOPTPS except that a given process is always displayed at the same position on the screen, that is, the list is not sorted by TPS.

5.2.34 Monitor Traffic

```
RTR> Monitor Traffic

                  LINK TRAFFIC  7-JAN-1999 14:38:08, NODE: -ALL-

                     Bytes/second     Packets/sec   Messages/sec    Congestion
                    Rcvd     Sent  Rcvd  Sent  Avrg  Rcvd   Sent    Abs   Rate
Total             13465.5  13213.3 123.8 121.2 239.1  53.3   53.3       35  0.0

NODEA -> NODEA        0.0      0.0   0.0   0.0   0.0   0.0    0.0        0  0.0
NODEA -> NODEB    10718.0   2705.8  95.1  27.5 118.9  24.9   27.2       28  0.0
NODEB -> NODEB       41.7     41.7   1.2   1.2   1.4   1.2    1.2        0  0.0
NODEB -> NODEA     2705.8  10465.8  27.5  92.5 118.8  27.2   24.9        7  0.0
```

Displays a list of the links to other nodes. Shown for each link are: byte rate, packet rate, message rate and congestion, in both directions. Average packets per second is also shown. Uses counters in the Network I/O (NIO) subsystem.

5.2.35 Monitor Trans

```
RTR> Monitor Trans

monitoring transactions Thu Mar  4 1999 15:21:38, NODE: NODEA
Tid  (Frontend)                    Facility                       FE-User
46d01f10,0,0,0,0,baa09da,abf10001  RTR$DEFAULT_FACILITY           RTRCSV_TETRAUL
aborting


Tid  (Router)                      Facility                       FE-User
 State


Tid  (Backend)                     Facility                       FE-User
46d01f10,0,0,0,0,baa09da,abf10001  RTR$DEFAULT_FACILITY           RTRCSV_TETRAUL
aborting
```

Displays transaction state for transactions in a facility.

5.2.36 Monitor V2CALLS

```
RTR> Monitor V2CALLS

RTR system service calls, Node: NODEA , PID: 00000000, Process name: -ALL-
Image: -ALL-                                                 13:09:18  5-MAR-1999
                     Accept    Reject   Success   Failure   Outstng     Calls
dcl_tx_prc/server         3         0         4         0         0         4
dcl_tx_prc/req.           1
dcl_tx_prc/shut.          0

start_tx                  1         0         1         0         0         1
start_tx /timeout         0

enq_tx                    1         0     50395         0         0     50395
enq_tx /broadcast     25187
enq_tx /reply         25207

deq_tx                50381         0     50391         0         2     50393
deq_tx /reply            12

commit_tx                 1         0         1         0         0         1

abort_tx                  0         0         0         0         0         0

vote_tx /commit       25187         0     25189         0         0     25189
vote_tx /abort            0
vote_tx /forget           2
```

Displays RTR Version 2 calls when running RTR Version 2 or mixture of RTR Version 3 and 2 environment.

5.2.37 Monitor XA

```
RTR> Monitor XA

      MONITOR XA
      RTR XA Calls    Node: nodea.zko          PID: -ALL-      Process name: -ALL-
      Image: -ALL-                                   11:42:06  Tue 6-Apr-1999
                      Calls      Success      Readonly        Failure
         open         0               0               0               0
         close        0               0               0               0
         start        0               0               0               0
         end          0               0               0               0
         prepare      0               0               0               0
         commit       0               0               0               0
         rollback     0               0               0               0
         recovery     0               0               0               0
                     Rate      0   2   4   6   8   10   12   14   16   18   20     Active
         Txn starts     0.0                                                0
```

Displays XA calls when using XA with RTR.

6
RTR Command Line Interface

Each RTR API call can be invoked at CLI level using the RTR command utility. This is provided to facilitate testing. For example, clients may be tested before the corresponding servers have been written by manually entering the server's API calls.

6.1 Introduction

The commands that invoke the RTR API calls are similar to the call names. For example, the rtr_accept_tx( ) call is invoked using CALL RTR_ACCEPT_TX at the CLI-level.

Where possible, command qualifiers have been given the same names as the parameters to the API calls. See the *Application Programmer's Reference Manual* for details about the parameters to API calls.

Most commands can be issued on remote nodes by using the /NODE=node-list or /CLUSTER qualifiers, or by preceding them with the SET ENVIRONMENT command to specify nodes where commands are to be executed. Commands such as DEFINE KEY are intended for local execution only.

Output from each command can be redirected to another device or file using the /OUTPUT qualifier.

Because the RTR command utility keeps parameter checking to a minimum, "what if" questions can be answered quickly without having to write test programs.

Note

In a mixed RTR Version 2 and Version 3 environment, you cannot execute commands remotely with the /NODE qualifier.

6.2 RTR Command Reference

This section describes in detail each command in the RTR command utility.

The command descriptions are presented in alphabetical order.

ADD FACILITY

See CREATE FACILITY; ADD FACILITY is retained for compatibility reasons only.

CALL RTR_ACCEPT_TX

The **CALL RTR_ACCEPT_TX** command causes a command server to execute the rtr_accept_tx( ) routine and to display the returned status.

Format

CALL RTR_ACCEPT_TX

Command Qualifiers	Defaults
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLUSTER	/NOCLUSTER
/FORGET	/NOFORGET
/INDEPENDENT	NOINDEPENDENT
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/REASON[=reason]	/REASON=0

Description

The **CALL RTR_ACCEPT_TX** command causes a command server to call the rtr_accept_tx( ) routine using values supplied on the command line.

The numeric status returned from the call is then converted to its textual representation and displayed.

The rtr_accept_tx( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_accept_tx( ) is:

```
rtr_status_t   rtr_accept_tx (
                   rtr_channel_t   channel,
                   rtr_acc_flag_t  flags,
                   rtr_reason_t    reason
                   ) ;
```

Table 6–1 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–1 Parameters for rtr_accept_tx

C Parameter Name	C Parameter Value	Command Line Specification
channel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	[none] [D]
	RTR_F_ACC_FORGET	/FORGET
reason	RTR_NO_REASON	/NOREASON [D]
	reason_value	/REASON=reason_value

Issuing the **CALL RTR_ACCEPT_TX** command in preference to using the /ACCEPT qualifier with the **CALL RTR_SEND_TO_SERVER** or **CALL RTR_REPLY_TO_CLIENT** commands is only necessary when you wish to specify an acceptance "reason" other than the default value of zero (using the /REASON qualifier).

Qualifiers

/CHANNEL_NAME=channel_name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the channel for which the operation is to be performed.

The command server uses a combination of the channel_name and the window from which the call was issued to uniquely identify which channel to use.

channel_name is not case sensitive.

The default channel name is RTR$DEFAULT_CHANNEL.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FORGET
/NOFORGET
Use /FORGET to specify the flags parameter as RTR_F_ACC_FORGET in the call rtr_accept_tx( ).

The default for value for /FORGET is /NOFORGET, which causes the command server to use the value RTR_NO_FLAGS for the flags parameter in the call to rtr_accept_tx( ).

/INDEPENDENT
NOINDEPENDENT
Use the /INDEPENDENT qualifier to specify the flags parameter RTR_F_ACC_ INDEPENDENT in the call to rtr_accept_txt( ).

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/REASON[=reason]
/REASON=0
Use /REASON to supply a value for the reason parameter in the call to rtr_accept_tx( ).

The default for value for /REASON is 0, which causes the command server to use the value RTR_NO_REASON for the reason parameter in the call to rtr_accept_tx( ).

Related Commands

- CALL RTR_OPEN_CHANNEL
- CALL RTR_REJECT_TX

Examples

Accept the current transaction with a reason of 42.

```
RTR> CALL RTR_ACCEPT_TX /REASON=42
%RTR-S-OK, normal successful completion
```

CALL RTR_BROADCAST_EVENT

The **CALL RTR_BROADCAST_EVENT** command causes a command server to execute the rtr_broadcast_event( ) routine and to display the returned status.

Format

CALL RTR_BROADCAST_EVENT [message-field1] [,message-field2...]

Parameters

[message-field1] [,message-field2...]
Specify the message to be sent (if any) as one or more comma separated parameter values. You can use the /TYPE_OF_DATA and /LENGTH_OF_DATA positional qualifiers on each parameter value to specify the data type and length of each field.

Command Qualifiers	Defaults
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLUSTER	/NOCLUSTER
/EVENT_NUMBER=evtnum	none
/FORMAT=fmt-string	/NOFORMAT
/LENGTH_OF_FIELD=message length	Depends on data type; see description.
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/RECIPIENT_SPEC=rcpspc	/NORECIPIENT_SPEC
/TYPE_OF_DATA=data type	/TYPE_OF_DATA=STRING

Description

The **CALL RTR_BROADCAST_EVENT** command causes a command server to call the rtr_broadcast_event( ) routine using values supplied on the command line.

The numeric status returned from the call is then converted to its textual representation and displayed.

The rtr_broadcast_event( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_broadcast_event( ) is:

```
rtr_status_t rtr_broadcast_event (
             rtr_channel_t     channel,
             rtr_bro_flag_t    flags,
             rtr_msgbuf_t      pmsg,
             rtr_msglen_t      msglen,
             rtr_evtnum_t      evtnum,
             rtr_rcpspc_t      rcpspc,
             rtr_msgfmt_t      msgfmt
             ) ;
```

Table 6–2 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–2 Parameters for rtr_broadcast_event

C Parameter Name	C Parameter Value	Command Line Specification
channel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	[none] [D]
pmsg, msglen, msgfmt[1]		[message definition parameter list with positional qualifiers.]
evtnum	42	/EVENT_NUMBER=42
rcpspc	"workstat*"	/RECIPIENT_SPEC="workstat*"

[1] The actual values used for pmsg, msglen and msgfmt are based upon the message definition you specify as a command line parameter.

The command server uses message data specified as command line parameter values to generate a record containing the message data (for the pmsg parameter), the message length (for the msglen parameter), and a record type description (for the msgfmt parameter).

Qualifiers

/CHANNEL_NAME=channel-name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the channel for which the operation is to be performed.

The command server uses a combination of the channel_name and the window from which the call was issued to uniquely identify which channel to use.

channel_name is not case sensitive.

The default for channel-name is RTR$DEFAULT_CHANNEL.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/EVENT_NUMBER=user-event-number
The user event number associated with this broadcast, in the range of RTR_EVTNUM_USERBASE to RTR_EVTNUM_USERMAX (i.e. 0 to 250).

/FORMAT[=fmt-string]
/NOFORMAT (D)
Specifies that a format string should be sent with this message.

If /FORMAT is specified without fmt-string, RTR automatically generates a format string. The format string is generated using the parameters given for the qualifiers /SIGNED, /UNSIGNED, /STRING and /LENGTH. The following table shows permitted values for these qualifiers when using /FORMAT without fmt-string.

Table 6–3 Generated Format Strings

Data Type	With /LENGTH=	With /NOLENGTH
STRING	=n, "%nC"	"%nC" where n=strlen(string)
SIGNED	=1, "%SB"	"%SL"
SIGNED	=2, "%SW"	"%SL"
SIGNED	=4, "%SL"	"%SL"
UNSIGNED	=1, "%UB"	"%SL"
UNSIGNED	=2, "%UW"	"%SL"
UNSIGNED	=4, "%UL"	"%SL"

Refer to *Application Programmer's Reference Manual*, section "Defining a Message Format Description" for information on constructing a `fmt-string` parameter.

/LENGTH_OF_FIELD=field-length
Enter the size of the message field that you want to define. The default for string types is the length of the message entered, plus one (for the zero termination byte). The default for signed and unsigned types is four. This is a positional qualifier; it must immediately follow the message field that it refers to.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in `node-list`. If `node-list` is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file `file-spec`. If `/OUTPUT` or `file-spec` is omitted then the standard or default output is used.

/RECIPIENT_SPEC=rcpspc
Enter a string specifying the recipient name. The wild-card characters "*" and "?" are permitted.

RECIPIENT_SPEC is case sensitive

/TYPE_OF_DATA=STRING|SIGNED|UNSIGNED
/TYPE_OF_DATA=STRING (D)
Enter the data type of the message field that you want to define. The default is the string type. This is a positional qualifier; it must immediately follow the message field that it refers to.

Examples

This command broadcasts user event number 23 to all channels having a null string rcpnam (the default). A message is sent with the broadcast.

```
RTR> CALL RTR_BROADCAST_EVENT "Dollar is up"/EVENT_NUMBER=23
%RTR-S-OK, Normal successful completion
```

The following command broadcasts user event number 24 to all recipients whose /RECIPIENT_NAME matches the DEALER% string (that is, DEALER1, DEALER2, DEALERx). Note that only the event is broadcast, there is no associated message.

```
RTR> CALL RTR_BROADCAST_EVENT /EVENT=24/RECIPIENT_SPEC=DEALER%
%RTR-S-OK, Normal successful completion
```

The following example shows a broadcast message containing two fields. The first field is of type unsigned, entered as a hexadecimal number; the second field is of type string.

```
RTR> CALL RTR_BROADCAST_EVENT /EVENT=24 0xFA9BC0 /TYPE_OF_
DATA=UNSIGNED/LENGTH=8,"This field of the message is a string"
%RTR-S-OK, Normal successful completion
```

CALL RTR_CLOSE_CHANNEL

The **CALL RTR_CLOSE_CHANNEL** command causes a command server to execute the rtr_close_channel( ) routine and to display the returned status.

Format

CALL RTR_CLOSE_CHANNEL

Command Qualifiers	Defaults
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLUSTER	/NOCLUSTER
/IMMEDIATE	/IMMEDIATE=RTR$DEFAULT_CHANNEL
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The **CALL RTR_CLOSE_CHANNEL** command causes a command server to call the rtr_close_channel( ) routine using values supplied on the command line.

The numeric status returned from the call is then converted to its textual representation and displayed.

The rtr_close_channel( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_close_channel( ) is:

```
rtr_status_t     rtr_close_channel (
                   rtr_channel_t       channel,
                   rtr_clo_flag_t      flags
                   ) ;
```

Table 6–4 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–4 Parameters for rtr_close_channel

C Parameter Name	C Parameter Value	Command Line Specification
channel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	[none] [D]

Qualifiers

/CHANNEL_NAME=channel_name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the channel for which the operation is to be performed.

The command server uses a combination of the channel_name and the window from which the call was issued to uniquely identify which channel to use.

channel_name is not case sensitive.

The default channel name is RTR$DEFAULT_CHANNEL.

You may close all the channels belonging to a window using CLOSE CHANNEL/CHANNEL_NAME=*.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/IMMEDIATE
/IMMEDIATE=RTR$DEFAULT_CHANNEL
Specifies the closing of a channel immediately without sending of a transaction acknowledgement.

Use /IMMEDIATE=RTR_F_CLO_IMMEDIATE to close the channel and recover any accepted transactions on this channel to an alternate server channel.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Examples

This command closes the RTR$DEFAULT_CHANNEL.

```
RTR> CALL RTR_CLOSE_CHANNEL
%RTR-S-OK, Normal successful completion
```

This command closes the channel named “client1”.

```
RTR> CALL RTR_CLOSE_CHANNEL/CHANNEL_NAME=CLIENT1
%RTR-S-OK, Normal successful completion
```

CALL RTR_ERROR_TEXT

The **CALL RTR_ERROR_TEXT** command causes a command server to execute the rtr_error_text( ) routine and to display the returned error text.

Format

CALL RTR_ERROR_TEXT

Command Qualifiers	Defaults
/OUTPUT[=file-spec]	/OUTPUT=stdout
/STATUS=status-code	/none

Description

The **CALL RTR_ERROR_TEXT** command causes a command server to call the rtr_error_text( ) routine using the value supplied on the command line.

The rtr_error_text( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_error_text( ) is:

```
char    *rtr_error_text (
           rtr_status_t    sts
           ) ;
```

Table 6–5 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–5 Parameters for rtr_error_text

C Parameter Name	Parameter Value	Command Line Specification
sts	42	42 (parameter)

Qualifiers

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/STATUS=status-code
No default
Specifies the sts parameter in the rtr_request_info( ) call.

Examples

This command shows the text associated with error number 4849722.

```
RTR> CALL RTR_ERROR_TEXT /STATUS=4849722
error text        normal successful completion
```

CALL RTR_GET_TID

The **CALL RTR_GET_TID** command causes a command server to execute the rtr_get_tid( ) routine and to display the returned status.

Format

CALL RTR_GET_TID

Command Qualifiers	Defaults
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLUSTER	/NOCLUSTER
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The **CALL RTR_GET_TID** command causes a command server to call to the rtr_get_tid( ) routine using values supplied on the command line.

The transaction id returned from the call is converted to its textual representation and displayed.

The rtr_get_tid( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_get_tid( ) is:

```
rtr_status_t    rtr_get_tid (
                   rtr_channel_t       channel,
                   rtr_tid_flag_t      flags,
void*              ptid
                   ) ;
```

Table 6–6 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–6 Parameters for rtr_get_tid

C Parameter Name	Parameter Value	Command Line Specification
channel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	[none] [D]

Qualifiers

/CHANNEL_NAME=channel_name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the channel for which the operation is to be performed.

The command server uses a combination of the channel_name and the window from which the call was issued to uniquely identify which channel to use.

The default channel name is RTR$DEFAULT_CHANNEL.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Examples

This command shows the transaction id active on channel "CLIENT1".

```
 RTR> CALL RTR_GET_TID/CHANNEL=CLIENT1
%RTR-S-OK, normal successful completion
 tid:           e100b810,0,0,0,0,3bc5,6eb02001
```

CALL RTR_OPEN_CHANNEL

The **CALL RTR_OPEN_CHANNEL** command causes a command server to execute the rtr_open_channel( ) routine and to display the returned status.

Format

CALL RTR_OPEN_CHANNEL

Command Qualifiers	Defaults
/ACCEPT_EXPLICIT	/NOACCEPT_EXPLICIT
/ACCESS=access	/NOACCESS
/BE_CALL_OUT	/NOBE_CALL_OUT
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLIENT	/NOCLIENT
/CLUSTER	/NOCLUSTER
/CONCURRENT	/CONCURRENT
/EVENTS[=event-nr-list]	/NOEVENTS
/FACILITY_NAME[=facility-name]	/FACILITY_NAME=RTR$DEFAULT_FACILITY
/FOREIGN_TM[=tm_id]	/NOFOREIGN_TM
/HIGH_BOUND=high-bound	/HIGH_BOUND=max-value-for-key-type
/KEY1=keysegdesc	
/KEYn=keysegdesc	
/LENGTH_OF_FIELD=key-field-length	/LENGTH_OF_FIELD=4
/LOW_BOUND=low-bound	/LOW_BOUND=lowest value for key-type
/NODE[=node-list]	/NODE=default-node-list
/OFFSET_OF_KEY=offset	/OFFSET_OF_KEY=0
/OUTPUT[=file-spec]	/OUTPUT=stdout
/PARTITION_NAME=partition-name	
/PREPARE_EXPLICIT	/NOPREPARE_EXPLICIT
/RECIPIENT_NAME=rcpnam	/RECIPIENT_NAME=null
/SERVER	/NOSERVER
/SHADOW	/NOSHADOW
/STANDBY	/STANDBY
/TR_CALL_OUT	/NOTR_CALL_OUT
/TYPE_OF_FIELD=key-field-type	/TYPE_OF_FIELD=UNSIGNED

Description

The **CALL RTR_OPEN_CHANNEL** command causes a command server to call the rtr_open_channel( ) routine using values supplied on the command line.

The numeric status returned from the call is then converted to its textual representation and displayed.

The rtr_open_channel( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_open_channel( ) is:

```
rtr_status_t     rtr_open_channel  (
                    rtr_channel_t       *pchannel,
                    rtr_ope_flag_t      flags,
                    rtr_facnam_t        facnam,
                    rtr_rcpnam_t        rcpnam,
                    rtr_evtnum_t        *pevtnum,
                    rtr_access_t        access,
                    rtr_numseg_t        numseg,
                    rtr_keyseg_t        *pkeyseg
                    ) ;
```

Table 6–7 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–7 Parameters for rtr_open_channel

C Parameter Name	C Parameter Value	Example
channel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	none [D]
	RTR_F_OPE_CLIENT	/CLIENT
	RTR_F_OPE_SERVER	/SERVER
	RTR_F_OPE_BE_CALL_OUT	/BE_CALL_OUT
	RTR_F_OPE_NOCONCURRENT	/NOCONCURRENT
	RTR_F_OPE_EXPLICIT_PREPARE	/PREPARE_EXPLICIT
	RTR_F_OPE_EXPLICIT_ACCEPT	/ACCEPT_EXPLICIT
	RTR_F_OPE_FOREIGN_TM	/FOREIGN=tm_id
	RTR_F_OPE_SHADOW	/SHADOW
	RTR_F_OPE_NOSTANDBY	/NOSTANDBY
	RTR_F_OPE_TR_CALL_OUT	/TR_CALL_OUT
facnam	"CASHFACIL"	/FACILITY_NAME=CASHFACIL
rcpnam	"CLI6CHAN"	/RECIPIENT_NAME=CLI6CHAN
pevtnum	[All events. See /EVENTS]	/EVENTS=(USER,RTR)
access	J67TF098	/ACCESS=J67TF098
numseg	1	
pkeyseg		
-> ks_type	rtr_keyseg_string	/TYPE_OF_FIELD=STRING
-> ks_length	10	/LENGTH_OF_FIELD=10
-> ks_offset	2	/OFFSET_OF_KEY=2
-> ks_lo_bound	"AAAAAAAAAA"	/LOW_BOUND="AAAAAAAAAA"
-> ks_hi_bound	"NNNNNNNNNN"	/HIGH_BOUND="NNNNNNNNNN"

Qualifiers

/ACCEPT_EXPLICIT
/NOACCEPT_EXPLICIT
Specifies that the RTR_F_OPE_EXPLICIT_ACCEPT flag is set in the call to `rtr_open_channel( )`.

/ACCESS=access
/NOACCESS (D)
Specifies an access string (that is, a password). All application programs (clients and servers) must specify the same access string for a given facility.

/BE_CALL_OUT
/NOBE_CALL_OUT (D)
Specifies that the RTR_F_OPE_BE_CALL_OUT flag is set in the flags parameter in the call to rtr_open_channel( ). The channel is opened as a backend call-out server.

/CHANNEL_NAME=channel-name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the name of the window's channel for use in subsequent operations on this channel.

channel_name is not case sensitive. The default channel name is RTR$DEFAULT_CHANNEL.

/CLIENT
/NOCLIENT
Specifies that the RTR_F_OPE_CLIENT flag is set on the call to rtr_open_channel( ). The channel is opened as a client.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/CONCURRENT (D)
/NOCONCURRENT
Specifying /NOCONCURRENT sets the RTR_F_OPE_NOCONCURRENT flag in the call to rtr_open_channel( ), and the server may not be concurrent with other servers. By default a server may have other concurrent servers.

/EVENTS=event-nr-list
/NOEVENTS (D)
The /EVENTS qualifier specifies that broadcast events are received on the channel. To subscribe to all user and RTR events, enter the qualifier with no arguments. Enter /EVENTS=RTR to receive the full range of RTR events only. Enter /EVENTS=USER to receive the full range of USER events only. Specify particular ranges of event numbers using arguments in the following format:

/EVENTS=(RTR, n, TO, m, USER, p, TO, q)

where n, m, p and q are event numbers. The default is to listen to no events.

/FACILITY_NAME=facility-name
/FACILITY=RTR$DEFAULT_FACILITY (D)
Specifies the name of the facility for which the channel is declared. An application must specify the facility name when using the RTR CLI. The default facility name is RTR$DEFAULT_FACILITY.

/FOREIGN_TM[=tm_id]
/NOFOREIGN_TM (D)
Valid for client channels only. This indicates that the global coordinating Transaction Manager (TM) is a Foreign TM (denoted as FTM), and that all TXs on this channel will be coordinated by the FTM. If this qualifier is set, then calls to rtr_start_tx on this channel must supply a value for the jointxid parameter, which is the TXID of the TX.

A TM identifier can also be passed in as parameter. It must be in the range of 0 to 65535. Default value is 0.

It is recommended that operators or script programs using nested transactions specify a TM identifier, particularly if more than one process opens RTR client channels using the same FTM on the one node, or if different types of FTMs are used on the same node. When a process that has open FTM client channels fails, then the FTM must be able to find out from RTR what state the transactions are in that were active in that process. Thus the FTM must be able to identify itself to RTR in order that RTR can find out what transactions were active for that FTM channel. Generally, FTM client channels opened in the same process (and for the same FTM) can have a common TM identifier, but FTM client channels opened in separate processes should have different TM identifiers.

Calling CALL RTR_OPEN_CHANNEL with the FOREIGN_TM qualifier specified will cause a local journal scan to occur if a journal has not already been opened on that node.

/HIGH_BOUND=high-bound
/HIGH_BOUND=max-val-for-key-type (D)
Specifies the upper bound of the key range that the server handles. The interpretation of `high-bound` depends on the key type. If the key is of type string then it is interpreted as text, otherwise it is interpreted as a numeric value. The default for `high-bound` is the largest possible value that can fit in the specified key type.

If the key bound value length is less than the key length (given in /LENGTH_OF_ FIELD), the key bound will automatically be null-padded to the required length.

/KEYn=keysegdesc
Specifies a partition key segment. Up to eight key segments may be defined for a partition (KEY1, KEY2,... up to KEY8).

The syntax of the KEYn qualifier is:

```
/KEYn=  (type_of_key=[signed|unsigned|string], -
  length_of_key=nnnn, -
  offset_of_key=nnnn, -
  low_bound=low-bound-string, -
  high_bound=high_bound_string)
```

`type_of_key=` Specifies the field type of the key. The `key-type` must be one of `unsigned`, `signed` or `string`. The default is `unsigned`.

length_of_key=nnnn Specifies the length of the key field in enqueued messages in bytes. Use this qualifier only if the key field type is string, since the key length is in other cases implied by the key type. The default value for key-length is four bytes.

offset_of_key=nnnn Specifies the offset of the key within the messages in bytes. The default is zero, that is, the key is at the start of the messages.

low_bound= Specifies the lower bound of the key range that servers in the partition will service. The interpretation of low-bound depends on the key type; if the key is of type string then it is interpreted as text, otherwise it is interpreted as a numeric value. The default for low-bound is the smallest possible value that can fit in the specified key type.

If the key bound value length is less than the key length (given in length_of_key), the key bound will automatically be null-padded to the required length.

high_bound= Specifies the upper bound of the key range that servers in the partition will service. The interpretation of high-bound depends on the key type. If the key is of type string then it is interpreted as text, otherwise it is interpreted as a numeric value. The default for high-bound is the largest possible value that can fit in the specified key type.

If the key bound value length is less than the key length (given in length_of_key), the key bound will automatically be null-padded to the required length.

Note

The /KEYn=keysegdesc qualifier is an alternative to use of the /xxx_OF_FIELD qualifier.

/LENGTH_OF_FIELD=key-field-length
/LENGTH_OF_FIELD=4 (D)
Specifies the length of the key field in the enqueued messages in bytes. Use this qualifier only if the key field type is string, since the key length is in all other cases implied by the key type. The default value for key-length is four bytes.

/LOW_BOUND=low-bound
/LOW_BOUND=min-val-for-key-type (D)
Specifies the lower bound of the key range that the server handles. The interpretation of low-bound depends on the key type. If the key is of type string then it is interpreted as text, otherwise it is interpreted as a numeric value. The default for low-bound is the smallest possible value that can fit in the specified key type.

If the key bound value length is less than the key length (given in /LENGTH_OF_FIELD), the key bound will automatically be null-padded to the required length.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OFFSET_OF_KEY=offset
/OFFSET_OF_KEY=0 (D)
Specifies the offset of the key within the messages in bytes. The default is zero, that is, the key is at the start of the messages.
Note that only one key segment definition is allowed.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file `file-spec`. If `/OUTPUT` or `file-spec` is omitted then the standard or default output is used.

/PARTITION_NAME=partition-name
Enables a system manager to specify a particular named partition.

/PREPARE_EXPLICIT
/NOPREPARE_EXPLICIT
Specifies that the flag RTR_F_OPE_EXPLICIT_PREPARE is set in the call to `rtr_open_channel( )`.

/RECIPIENT_NAME=rcpnam
/RECIPIENT_NAME=null (D)
Specifies an optional name for broadcast reception to be associated with the channel. Only broadcasts matching this name are delivered on the channel. (Note that to receive named events, the correct event number must also be specified.)

RECIPIENT_NAME is case sensitive. If /RECIPIENT_NAME is supplied without a qualifier value, then its value defaults to the value of /CHANNEL_NAME. Not supplying the /RECIPIENT_NAME qualifier has the same effect as specifying a null string. Wild card characters `cannot` be used for this qualifier. (Senders of broadcasts can use wildcards when specifying /RECIPIENT_SPEC).

/SERVER
/NOSERVER (D)
Specifies that the RTR_F_OPE_SERVER flag is set in the call to `rtr_open_channel( )`. Use this qualifier if you want to declare the channel as a server.

/SHADOW
/NOSHADOW (D)
The /SHADOW qualifier specifies that the RTR_F_OPE_SHADOW flag is set in the call to `rtr_open_channel( )`. The server is part of a shadow pair.

/STANDBY (D)
/NOSTANDBY
The /NOSTANDBY qualifier specifies that the flag RTR_F_OPE_NOSTANDBY is set in the call to `rtr_open_channel( )`, and the server may not be (or have) standby(s). By default, servers may have standby(s).

/TR_CALL_OUT
/NOTR_CALL_OUT (D)
Specifies that the RTR_F_OPE_TR_CALL_OUT flag is set in the call to `rtr_open_channel( )`, and the server is a router call-out server. By default a server is not a router call-out server.

/TYPE_OF_FIELD=key-field-type
/TYPE_OF_FIELD=UNSIGNED (D)
Specifies the field type of the key. The key-type must be one of UNSIGNED, SIGNED or STRING. The default is UNSIGNED.

Related Commands

- CALL RTR_CLOSE_CHANNEL

Examples

This command opens a server channel called RTR$DEFAULT_CHANNEL that may not have concurrent servers, explicitly accepts transactions and listens for all RTR events.

```
RTR> CALL RTR_OPEN_CHANNEL/SERVER/NOCONCURRENT/ACCEPT_EXPLICIT/EVENTS=RTR
%RTR-S-OK, Normal successful completion
```

This command open a client channel called “FIN1CHAN”.

```
RTR> CALL RTR_OPEN_CHANNEL/CLIENT/CHANNEL=FIN1CHAN
%RTR-S-OK, Normal successful completion
```

CALL RTR_PREPARE_TX

The **CALL RTR_PREPARE_TX** command causes a command server to execute the rtr_prepare_tx( ) routine and to display the returned status.

Format

CALL RTR_PREPARE_TX

Command Qualifiers	Defaults
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/DATA[=data]	/DATA=0
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/REASON[=reason]	/REASON=0

Description

The **CALL RTR_PREPARE_TX** command causes a command server to call the rtr_prepare_tx( ) routine using values supplied on the command line.

The numeric status returned from the call is then converted to its textual representation and displayed.

The rtr_prepare_tx( ) routine itself is described in *Application Programmer's Reference Manual.*

The prototype of rtr_prepare_tx( ) is:

```
rtr_status_t   rtr_prepare_tx (
                    rtr_channel_t   channel,
                    rtr_pre_flag_t  flags,
                    rtr_reason_t    reason
     rtr_msgbuf_t    pmsg
     rtr_msglen_t    msglen
                    ) ;
```

Table 6–8 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–8 Parameters for rtr_prepare_tx

C Parameter name	C Parameter Value	Command Line Specification
channel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	[none] [D]
reason	RTR_NO_REASON	/NOREASON [D]
	reason_value	/REASON=reason_value
pmsg, msglen		/ DATA=data

The CALL RTR_PREPARE_TX command causes a command server to execute the rtr_prepare_tx() routine using the values supplied on the command line. The numeric status returned from the call is then converted to its textual representation and displayed.

The CALL RTR_PREPARE_TX can only be used in the context of nested transactions (rtr_start_tx was called with the parameter join_txid not equal to NOJOIN_TXID). If this call returns RTR_STS_OK, then the first (prepare) phase of the RTR 2PC protocol has been initiated.

The message type associated with this command is rtr_mt_prepared. Similar to the message types rtr_mt_accepted and rtr_mt_rejected, the rtr_mt_prepared message type returns data of type rtr_status_data_t in the user buffer. In this case, the status field of rtr_status_data_t is always RTR_STS_OK, and the reason field contains the same reason mask that would be returned in the rtr_mt_accepted message type for the same TX were the TX to be accepted.

Only when the rtr_mt_prepared message is delivered, can the operator be sure that all participants of the nested TX are ready to commit. Alternatively, calling rtr_prepare_tx can result in the message rtr_mt_rejected being delivered if one of the participating servers votes to reject the nested TX.

The rtr_mt_prepared message is only delivered to the console if the rtr_prepare_tx is called.

A call to rtr_prepare_tx must be followed at some point by a call to rtr_accept_tx (or rtr_reject_tx), which in this context implements the commit phase of the 2PC. (Generally, rtr_prepare_tx and rtr_accept_tx will be called by the foreign TM directly, not by the operator).

Qualifiers

/CHANNEL_NAME=channel_name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the channel for which the operation is to be performed.

The command server uses a combination of the channel_name and the window from which the call was issued to uniquely identify which channel to use.

channel_name is not case sensitive.

The default channel name is RTR$DEFAULT_CHANNEL.

/DATA[=data]
/DATA=0
The data parameter can be used to pass an ASCII string to RTR that will be saved in the local journal. This string is not sent to the routers or backends, and is used only during recovery, when it is passed back to the client application. RTR does not interpret or modify this data in any way. The maximum size of the data parameter is defined as RTR_MAX_BLOB_LEN and is set to 2048 bytes.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/REASON[=reason]
/REASON=0
Use /REASON to supply a value for the reason parameter in the call to rtr_prepare_tx( ).

The reason parameter to rtr_prepare_tx( ) is used in place of the reason parameter in the subsequent rtr_accept_tx call( ) (that is, the reason field in the call to rtr_accept_tx call( ) or rtr_reject_tx( ) which follows a call to rtr_prepare_tx( ) is ignored). The default for value for /REASON is 0.

Related commands

- CALL RTR_OPEN_CHANNEL
- CALL RTR_REJECT_TX

Examples

Prepare the current transaction with a reason of 42.

```
RTR> CALL RTR_PREPARE_TX /REASON=42
%RTR-S-OK, normal successful completion
```

CALL RTR_RECEIVE_MESSAGE

The **CALL RTR_RECEIVE_MESSAGE** command causes a command server to execute the rtr_receive_message( ) routine and to display the returned status.

Format

CALL RTR_RECEIVE_MESSAGE

Command Qualifiers	Defaults
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLUSTER	/NOCLUSTER
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/TIMEOUT_MS=timoutms	/TIMEOUT_MS=infinite

Description

The **CALL RTR_RECEIVE_MESSAGE** command causes a command server to call the rtr_receive_message( ) routine using values supplied on the command line.

The numeric status returned from the call is then converted to its textual representation and displayed.

The rtr_receive_message( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_receive_message( ) is:

```
rtr_status_t  rtr_receive_message (
                rtr_channel_t       *pchannel,
                rtr_rcv_flag_t      flags,
                rtr_channel_t       *prcvchan,
                rtr_msgbuf_t        pmsg,
                rtr_msglen_t        maxlen,
                rtr_timout_t        timoutms,
                rtr_msgsb_t         *pmsgsb
                ) ;
```

Table 6–9 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–9 Parameters for rtr_receive_message

C Parameter Name	C Parameter Value	Command Line Specification
pchannel		[displayed]
flags	RTR_NO_FLAGS	[none] [D]
prcvchan		/CHANNEL=channel_name
pmsg		[message displayed, if any]
maxlen	RTR_MAX_MSGLEN	[reasonable limit for display]
timoutms		/TIMEOUT_MS=timoutms
pmsgsb		[relevant fields displayed]

For all messages received, RTR displays the contents of the message status block (msgsb) as follows:

the message type (for example, rtr_mt_opened, rtr_mt_msgn).
the message length in bytes
the transaction ID, user handle, are event number are shown if they are relevant for the message type.

For message types that place a status code and reason code in the user buffer, the status code is converted to text and both are shown.

For all other message types, the contents of the user buffer are displayed hexadecimal and ASCII text.

Qualifiers

/CHANNEL_NAME=channel_name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies one or more channels from which a message may be received. To specify two or more channels, enter a comma-separated list, e.g. /CHANNEL_NAME=(CHAN1,CHAN2,CHAN5). Channel names may include wildcard characters.

channel_name is not case sensitive.

Entering the qualifier without a value (that is, /CHANNEL) is equivalent to /CHANNEL=RTR$DEFAULT_CHANNEL. Omitting the qualifier altogether is equivalent to /CHANNEL=* (that is, receive a message on any defined channel).

The command server uses a combination of the channel-name and the window from which the call was issued to uniquely identify which channel to use.

Note that this qualifier sets up the prcvchan parameter on the call to rtr_receive_message.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/TIMEOUT_MS=timoutms
/TIMEOUT_MS=0 (D)
The **timoutms** argument defines a timeout for the receive, in milliseconds. The default value is an infinite timeout. /TIMEOUT_MS specifies an immediate timeout.

Examples

The following example shows two **CALL RTR_RECEIVE_MESSAGE** commands on RTR$DEFAULT_CHANNEL.

```
RTR> CALL RTR_RECEIVE_MESSAGE /TIMEOUT_MS
%RTR-S-OK, normal successful completion
channel name:  RTR$DEFAULT_CHANNEL
msgsb
  msgtype:     rtr_mt_rtr_event
  msglen:      0
  evtnum:      113     (RTR_EVTNUM_SRRECOVERCMPL)
RTR> CALL RTR_RECEIVE_MESSAGE /TIMEOUT_MS
%RTR-S-OK, normal successful completion
channel name:  RTR$DEFAULT_CHANNEL
msgsb
  msgtype:     rtr_mt_msg1
  msglen:      55
  usrhdl:  0
  tid:         1fe5c,495a4,0,0,1bf80,84,36
message
  offset  bytes                                           text
  000000  54 68 69 73 20 69 73 20 74 68 65 20 6D 65 73 73  This is the mess
  000010  61 67 65 20 74 65 78 74 2E 20 53 6F 6D 65 20 64  age text. Some d
  000020  69 67 69 74 73 20 68 65 72 65 3A 20 31 32 33 34  igits here: 1234
  000030  35 36 37 38 39 30 00                            567890.
```

CALL RTR_REJECT_TX

The **CALL RTR_REJECT_TX** command causes a command server to execute the rtr_reject_tx( ) routine and to display the returned status.

Format

CALL RTR_REJECT_TX

Command Qualifiers	Defaults
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLUSTER	/NOCLUSTER
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/REASON[=rsnsts]	/REASON=0
/RETRY	/NORETRY (D)

Description

The **CALL RTR_REJECT_TX** command causes a command server to call the rtr_reject_tx( ) routine using values supplied on the command line.

The numeric status returned from the call is then converted to its textual representation and displayed.

The rtr_reject_tx( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_reject_tx( ) is:

```
rtr_status_t   rtr_reject_tx (
                   rtr_channel_t   channel,
                   rtr_rej_flag_t  flags,
                   rtr_reason_t    reason
                   ) ;
```

Table 6–10 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–10 Parameters for rtr_reject_tx

C Parameter Name	C Parameter Value	Command Line Specification
channel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	[none] [D]
	RTR_F_REJ_RETRY	/RETRY
reason	RTR_NO_REASON	/NOREASON [D]
	reason_value	/REASON=reason_value

Qualifiers

/CHANNEL_NAME=channel_name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the channel for which the operation is to be performed.

The command server uses a combination of the channel_name and the window from which the call was issued to uniquely identify which channel to use.

channel_name is not case sensitive.

The default channel name is RTR$DEFAULT_CHANNEL.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/RETRY
/NORETRY (D)
Use /RETRY to specify the flags parameter as RTR_F_REJ_RETRY in the call rtr_reject_tx( ).

The default for value for /RETRY is /NORETRY, which causes the command server to use the value RTR_NO_FLAGS for the flags parameter in the call to rtr_reject_tx( ).

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/REASON[=reason]
/REASON=0
Use /REASON to supply a value for the reason parameter in the call to rtr_reject_tx( ).

The default for value for /REASON is 0, which causes the command server to use the value RTR_NO_REASON for the reason parameter in the call to rtr_reject_tx( ).

Related Commands

- CALL RTR_OPEN_CHANNEL
- CALL RTR_ACCEPT_TX

Examples

Reject the current transaction with a reason of 42.

```
RTR> CALL RTR_REJECT_TX /REASON=42
%RTR-S-OK, normal successful completion
```

CALL RTR_REPLY_TO_CLIENT

The **CALL RTR_REPLY_TO_CLIENT** command causes a command server to execute the rtr_reply_to_client( ) routine and to display the returned status.

Format

CALL RTR_REPLY_TO_CLIENT [message-field1] [,message-field2...]

Parameters

[message-field1] [,message-field2...]
Specify the message to be sent as one or more comma separated parameter values. You can use the /TYPE_OF_DATA and /LENGTH_OF_DATA positional qualifiers on each parameter value to specify the data type and length of each field.

Command Qualifiers	Defaults
/ACCEPT	NOACCEPT
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLUSTER	/NOCLUSTER
/FORMAT=fmt-string	/NOFORMAT
/INDEPENDENT	NOINDEPENDENT
/LENGTH_OF_FIELD=message length	Depends on data type; see description.
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/TYPE_OF_DATA=data type	/TYPE_OF_DATA=STRING

Description

The **CALL RTR_REPLY_TO_CLIENT** command causes a command server to call the rtr_reply_to_client( ) routine using values supplied on the command line.

The numeric status returned from the call is then converted to its textual representation and displayed.

The rtr_reply_to_client( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_reply_to_client( ) is:

```
rtr_status_t   rtr_reply_to_client (
                   rtr_channel_t       channel,
                   rtr_rep_flag_t      flags,
                   rtr_msgbuf_t        pmsg,
                   rtr_msglen_t        msglen,
                   rtr_msgfmt_t        msgfmt
                   ) ;
```

Table 6–11 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–11 Parameters for rtr_reply_to_client

C Parameter Name	C Parameter Value	Command Line Specification
channel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	none [D]
	RTR_F_REP_ACCEPT	/ACCEPT
pmsg, msglen, msgfmt[1]		[message definition parameter list with positional qualifiers.]

[1] The actual values used for pmsg, msglen and msgfmt are based upon the message definition you specify as a command line parameter.

The command server uses message data specified as command line parameter values to generate a record containing the message data (for the pmsg parameter), the message length (for the msglen parameter), and a record type description (for the msgfmt parameter).

Qualifiers

/ACCEPT
/NOACCEPT
The /ACCEPT qualifier sets the flag RTR_F_REP_ACCEPT in the call to reply_to_client( ). It means the transaction is accepted by this server.

/CHANNEL_NAME=channel_name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the channel for which the operation is to be performed.

The command server uses a combination of the channel_name and the window from which the call was issued to uniquely identify which channel to use.

channel_name is not case sensitive.

The default channel name is RTR$DEFAULT_CHANNEL.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FORMAT[=fmt-string]
/NOFORMAT (D)
Specifies that a format string should be sent with this message.

If /FORMAT is specified without fmt-string, RTR automatically generates a format string. The format string is generated using the parameters given for the qualifiers /SIGNED, /UNSIGNED, /STRING and /LENGTH. The following table shows permitted values for these qualifiers when using /FORMAT without fmt-string.

Table 6–12 Generated Format Strings

Data Type	With /LENGTH=	With /NOLENGTH
STRING	=n, "%nC"	"%nC" where n=strlen(string)
SIGNED	=1, "%SB"	"%SL"
SIGNED	=2, "%SW"	"%SL"
SIGNED	=4, "%SL"	"%SL"
UNSIGNED	=1, "%UB"	"%SL"
UNSIGNED	=2, "%UW"	"%SL"
UNSIGNED	=4, "%UL"	"%SL"

Refer to *Application Programmer's Reference Manual*, section "Defining a Message Format Description" for information on constructing a `fmt-string` parameter.

/INDEPENDENT
NOINDEPENDENT
Use the /INDEPENDENT qualifier to specify the flags parameter RTR_F_ACC_ INDEPENDENT in the call to rtr_reply_to_client( ).

/LENGTH_OF_FIELD=field-length
Enter the size of the message field that you want to define. The default for string types is the length of the message entered, plus one (for the zero termination byte). The default for signed and unsigned types is four. This is a positional qualifier; it must immediately follow the message field that it refers to.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in `node-list`. If `node-list` is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file `file-spec`. If `/OUTPUT` or `file-spec` is omitted then the standard or default output is used.

/TYPE_OF_DATA=STRING|SIGNED|UNSIGNED
/TYPE_OF_DATA=STRING (D)
Enter the data type of the message field that you want to define. The default is the string type. This is a positional qualifier; it must immediately follow the message field that it refers to.

Related commands

- `CALL RTR_SEND_TO_SERVER`
- `CALL RTR_RECEIVE_MESSAGE`

Examples

The following example replies a message to the client.

```
RTR>  CALL RTR_REPLY_TO_CLIENT "Getting that info for you"
 %RTR-S-OK, Normal successful completion
```

The following example shows a message of type unsigned and entered as a hexadecimal number.

```
 RTR> CALL RTR_REPLY_TO_CLIENT "0xFA9BC0"/TYPE_OF_DATA=UNSIGNED
 %RTR-S-OK, Normal successful completion
```

CALL RTR_REQUEST_INFO

The **CALL RTR_REQUEST_INFO** command causes a command server to execute the rtr_request_info( ) routine and to display the returned status.

Format

CALL RTR_REQUEST_INFO

Command Qualifiers	Defaults
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLUSTER	/NOCLUSTER
/GETITM=item-name	none
/INFCLA=infoclass	none
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/SELITM=item-name	none
/SELVAL=item-value	none

Description

The **CALL RTR_REQUEST_INFO** command causes a command server to call to the rtr_request_info( ) routine using values supplied on the command line.

The rtr_request_info( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_request_info( ) is:

```
    rtr_status_t    rtr_request_info (
rtr_channel_t       *pchannel,
rtr_req_flag_t      flags,
rtr_infoclass_t     infcla
rtr_itemcode_t      selitm
rtr_selval_t        selval
rtr_itemcode_t      getitms
) ;
```

Table 6–13 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–13 Parameters for rtr_request_info

C Parameter Name	Parameter Value	Command Line Specification
*pchannel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	[none] [D]
infcla	RTR_INFCLA_BACKEND_TX	/INFCLA=BTX
selitm	fdb_f_name	/SELITM=fdb_f_name
selval	CASHFACIL	/SELVAL=CASHFACIL
getitms	fe_node_id	/GETITMS=fe_node_id

Qualifiers

/CHANNEL_NAME=channel_name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the channel for which the operation is to be performed.

The command server uses a combination of the channel_name and the window from which the call was issued to uniquely identify which channel to use.

channel_name is not case sensitive.

The default channel name is RTR$DEFAULT_CHANNEL.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/GETITM=item-name[,item-name...]
No default
Specifies the getitm parameter in the rtr_request_info( ) call.

/INFCLA=infoclass
No default
Specifies the infcla parameter in the rtr_request_info( ) call.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/SELITM=item-name
No default
Specifies the selitm parameter in the rtr_request_info( ) call.

/SELVAL=item-value
No default
Specifies the selval parameter in the rtr_request_info( ) call.

Examples

This command requests the backend transaction IDs for the facility “CASHFAC”.

```
RTR> CALL RTR_REQUEST_INFO/CHANNEL=INFOCHAN
/INFCLA="btx"/SELITM=fac_id/SELVAL=CASHFAC
/GETITMS=tb_txdx.tx_id
```

The information can then be viewed by repeatedly executing the following command until the channel is closed.

```
RTR> CALL RTR_RECEIVE_MESSAGE/CHANNEL=INFOCHAN/TIMEOUT
```

CALL RTR_SEND_TO_SERVER

The **CALL RTR_SEND_TO_SERVER** command causes a command server to execute the rtr_send_to_server( ) routine and to display the returned status.

Format

CALL RTR_SEND_TO_SERVER [message-field1] [,message-field2...]

Parameters

[message-field] [,message-field2...]
Specify the message to be sent as one or more comma separated parameter values. You can use the /TYPE_OF_DATA and /LENGTH_OF_DATA positional qualifiers on each parameter value to specify the data type and length of each field.

Command Qualifiers	Defaults
/ACCEPT	NOACCEPT
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLUSTER	/NOCLUSTER
/EXPENDABLE	/NOEXPENDABLE
/FORMAT=fmt-string	/NOFORMAT
/LENGTH_OF_FIELD=message-length	Depends on data type; see description.
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/READONLY	/NOREADONLY
/RETURN_TO_SENDER	/NORETURN_TO_SENDER
/TYPE_OF_DATA=data type	/TYPE_OF_DATA=STRING

Description

The **CALL RTR_SEND_TO_SERVER** command causes a command server to call the rtr_send_to_server( ) routine using values supplied on the command line.

The numeric status returned from the call is then converted to its textual representation and displayed.

The rtr_send_to_server( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_send_to_server( ) is:

```
rtr_status_t   rtr_send_to_server (
                   rtr_channel_t       channel,
                   rtr_sen_flag_t      flags,
                   rtr_msgbuf_t        pmsg,
                   rtr_msglen_t        msglen,
                   rtr_msgfmt_t        msgfmt
                   ) ;
```

Table 6–14 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–14 Parameters for rtr_send_to_server

C Parameter Name	C Parameter Value	Command Line Specification
channel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	none [D]
	RTR_F_SEN_EXPENDABLE	/EXPENDABLE
	RTR_F_SEN_READONLY	/READONLY
	RTR_F_SEN_RETURN_TO_SENDER	/RETURN_TO_SENDER
pmsg, msglen, msgfmt[1]		[message definition parameter list with positional qualifiers.]

[1] The actual values used for pmsg, msglen and msgfmt are based upon the message definition you specify as a command line parameter.

The command server uses message data specified as command line parameter values to generate a record containing the message data (for the pmsg parameter), the message length (for the msglen parameter), and a record type description (for the msgfmt parameter).

Qualifiers

/ACCEPT
/NOACCEPT
The /ACCEPT qualifier sets the flag RTR_F_REP_ACCEPT in the call to send_to_server( ). It means the transaction is accepted by this client.

/CHANNEL_NAME=channel_name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the channel for which the operation is to be performed.

The command server uses a combination of the channel_name and the window from which the call was issued to uniquely identify which channel to use.

channel_name is not case sensitive.

The default channel name is RTR$DEFAULT_CHANNEL.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/EXPENDABLE
/NOEXPENDABLE
The /EXPENDABLE qualifier sets the flag RTR_F_SEN_EXPENDABLE in the call rtr_send_to_server( ).

/FORMAT[=fmt-string]
/NOFORMAT (D)
Specifies that a format string should be sent with this message.

If /FORMAT is specified without fmt-string, RTR automatically generates a format string. The format string is generated using the parameters given for the qualifiers /SIGNED, /UNSIGNED, /STRING and /LENGTH. The following table shows permitted values for these qualifiers when using /FORMAT without fmt-string.

Table 6–15 Generated Format Strings

Data Type	With /LENGTH=	With /NOLENGTH
STRING	=n, "%nC"	"%nC" where n=strlen(string)
SIGNED	=1, "%SB"	"%SL"
SIGNED	=2, "%SW"	"%SL"
SIGNED	=4, "%SL"	"%SL"
UNSIGNED	=1, "%UB"	"%SL"
UNSIGNED	=2, "%UW"	"%SL"
UNSIGNED	=4, "%UL"	"%SL"

Refer to *Application Programmer's Reference Manual*, section "Defining a Message Format Description" for information on constructing a fmt-string parameter.

/LENGTH_OF_FIELD=field-length
Enter the size of the message field that you want to define. The default for string types is the length of the message entered, plus one (for the zero termination byte). The default for signed and unsigned types is four. This is a positional qualifier; it must immediately follow the message field that it refers to.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/READONLY
/NOREADONLY
The /READONLY qualifier sets the flag RTR_F_SEN_READONLY in the call rtr_send_to_server( ).

/RETURN_TO_SENDER
/NORETURN_TO_SENDER
The /RETURN_TO_SENDER qualifier sets the flag RTR_F_SEN_RETURN_TO_SENDER in the call rtr_send_to_server( ).

/TYPE_OF_DATA=STRING|SIGNED|UNSIGNED
/TYPE_OF_DATA=STRING (D)
Enter the data type of the message field that you want to define. The default is the string type. This is a positional qualifier; it must immediately follow the message field that it refers to.

Examples

This command sends a message to a server. The message is type string (the default).

```
RTR>  CALL RTR_SEND_TO_SERVER "Get that info for me, please"
%RTR-S-OK, Normal successful completion
```

CALL RTR_START_TX

The **CALL RTR_START_TX** command causes a command server to execute the rtr_start_tx( ) routine and to display the returned status.

Format

CALL RTR_START_TX

Command Qualifiers	Defaults
/CHANNEL_NAME=channel-name	/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
/CLUSTER	/NOCLUSTER
/JOIN_TXID=txid-number	/NOJOIN_TXID
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/TIMEOUT_MS=timoutms	/TIMEOUT_MS=0
/TXID_TYPE=txid-type	/TXID_TYPE=RTR

Description

The **CALL RTR_START_TX** command causes a command server to call the rtr_start_tx( ) routine using values supplied on the command line.

The numeric status returned from the call is then converted to its textual representation and displayed.

The rtr_start_tx( ) routine itself is described in *Application Programmer's Reference Manual*.

The prototype of rtr_start_tx( ) is:

```
rtr_status_t   rtr_start_tx (
                   rtr_channel_t       channel,
                   rtr_sta_flag_t      flags,
                   rtr_timout_t        timoutms,
                   rtr_channel_t       joinchan
                   ) ;
```

Table 6–16 shows the correspondence between values you supply on the command line and the C language parameter values produced and used for the call.

Table 6–16 Parameters for rtr_start_tx

C Parameter Name	C Parameter Value	Command Line Specification
channel		/CHANNEL_NAME=name
flags	RTR_NO_FLAGS	[none] [D]
timoutms		/TIMEOUT_MS=timoutms
joinchan		/JOIN_CHANNEL=channel-name

Nested Transaction Usage

If the FOREIGN_TM qualifier is specified for channel, then the global coordinating TM for this transaction is a foreign TM. In this case, rtr_start_tx( ) must be called to start a transaction (cannot be started implicitly on first call to rtr_send_to_server( ) and the join_txid parameter must be specified. The txid_type parameter defaults to RTR if not specified. (This restriction, that classic transactions without a prepare

phase cannot be executed on an client FTM channel, may be relaxed in a future release.)

When a nested transaction is started (join_txid not equal to NOJOIN_TXID), then that transaction is given a new RTR TXID (which the operator can retrieve by calling rtr_get_tid). The foreign TXID passed in join_txid is used only to identify the transaction for the foreign TM (for example, when the foreign TM goes through recovery and requests RTR to return all transactions in prepared state).

The channel on which a nested transaction is started must be opened as a client channel, which implies the node is defined with the FE role. However, a journal is required on that node. If there are no other facilities on that node with the BE role defined, then RTR will behave as an honorary BE, causing the journal to be opened locally.

Qualifiers

/CHANNEL_NAME=channel_name
/CHANNEL_NAME=RTR$DEFAULT_CHANNEL
Specifies the channel for which the operation is to be performed.

The command server uses a combination of the channel_name and the window from which the call was issued to uniquely identify which channel to use.

channel_name is not case sensitive.

The default channel name is RTR$DEFAULT_CHANNEL.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/JOIN_TXID=txid-number
/NOJOIN_TXID
The format of the txid-number depends on the TXID_TYPE:

```
TXID_TYPE          Format of txid-number
-----------------------------------------------------------
RTR                7 integers of 4 bytes each, separated by commas
DDTM               4 integers of 4 bytes each, separated by commas
XA                 Character-string up to 128 bytes of length
```

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/TIMEOUT_MS=timoutms
/TIMEOUT_MS=0 (D)
The **timoutms** argument defines a timeout for the transaction, in milliseconds. The default value is an infinite timeout. /TIMEOUT_MS specifies an immediate timeout (timeout if no message to receive).

/TXID_TYPE=txid-type
/TXID_TYPE=RTR (D)
Possible values for TXID_TYPE parameter are RTR, DDTM and XA.

Related commands

- CALL RTR_REJECT_TX
- CALL RTR_ACCEPT_TX

Examples

This command starts a new transaction with a timout of 5000 milliseconds.

```
RTR>  CALL RTR_START_TX /TIMEOUT_MS=5000
%RTR-S-OK, Normal successful completion
```

CLEAR

Interactively remove one or more displayed items from a monitor picture.

Format

CLEAR

Command Qualifiers	Defaults
/ALL	/NOALL
/X=column	
/Y=row	

Description

The CLEAR command enables you to interactively remove one or all of the displayed items from a monitor picture. The picture can then be redisplayed using the MONITOR /RESUME command.

CLEAR /ALL removes all the displayed items from the screen, and is useful when commencing interactive definition of a new picture.

Either /ALL, or /X and /Y must be given.

Qualifiers

/ALL
/NOALL (D)
Specifies that all the items are to be removed from the monitor picture. This is usually used when commencing the interactive definition of a new picture.

/X=column
/Y=row
Specify that the item in position (column,row) is to be removed from the monitor picture. This enables mistakes to be corrected when interactively modifying monitor pictures.

It can also be used to remove unwanted items from predefined monitor pictures.

Related commands

- MONITOR
- SHOW DISPLAY
- DISPLAY NUMERIC
- DISPLAY BAR
- DISPLAY TEXT

Examples

See Section A.1, Interactive Definition of a Monitor Picture, for an example of how to use the CLEAR command.

CREATE FACILITY

Create an RTR facility and ready it for transaction traffic.

Format

CREATE FACILITY [facility_name]

Command Qualifiers	Defaults
/ALL_ROLES=node-list	/NOALL_ROLES
/BACKEND=backend-list	/NOBACKEND
/BALANCE	/NOBALANCE
/CALL_OUT=role-list	/NOCALL_OUT
/CLUSTER	/NOCLUSTER
/FRONTEND=frontend-list	/NOFRONTEND
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/REPLY_CHECKSUM	NOREPLY_CHECKSUM
/RESOURCE_MANAGER=resource-list	
/ROUTER=router-list	/NOROUTER

Description

The CREATE FACILITY command configures (defines on a node) an RTR facility and readies it for transaction traffic (that is, establishes links to other the participating nodes).

The command must be issued on all participating nodes before any application programs using the facility are started.

Note

Router nodes must have *all* backend nodes defined and Backend nodes must have *all* router nodes defined.

Routers need only be defined with the frontends that they can connect to.

Frontends need only be defined with the routers they can connect to.

Parameters

facility_name
Specifies the name of the facility to be created.

Any application program which uses this facility must specify the same name when it calls rtr_open_channel( ).

Facility names can contain up to thirty-one characters. Letters, numbers and underline characters are all valid, but the first character of a facility name must be a letter.

Facility names are not case sensitive.

The default value for facility_name is RTR$DEFAULT_FACILITY.

The /ROUTER qualifier, and at least one of /FRONTEND or /BACKEND must be specified.

Qualifiers

/ALL_ROLES=node-list
/NOALL_ROLES (D)
Specifies the names of the nodes that are to act as frontend, router and backend in this facility.

Note that the definition order of nodes may be significant. This applies to the order of router node definitions when frontend load balancing is not enabled. Nodes defined with the /ROUTER qualifier have the higher priority and are followed by nodes defined by the /ALL_ROLES qualifier. For example, in this definition:

```
$ RTR CREATE FACILITY /ALL_ROLES=mynode /ROUTER=(anode,bnode)
```

The router nodes are in definition order anode, bnode, mynode for all frontends except mynode. (Any node that has both frontend and router roles selects its own router first.)

/BACKEND=backend-list
/NOBACKEND (D)
Specifies the names of the nodes that are to act as backends for this facility.

Backend-list is a list of backend-nodes separated by commas. If there is more than one backend-node, then backend-list must be enclosed in parentheses.

Backend-node is either the name of a node or @file-spec, where file-spec specifies a text file containing a backend-list on each line.

/BALANCE
/NOBALANCE (D)
Specifies that load balancing is enabled for the frontend of the transaction router listed with /ROUTER. See Section 2.6, Router Load Balancing, for details on load balancing.

It has no significance on a backend node, and will be ignored if specified.

The default behavior (/NOBALANCE) is for a frontend to connect to the preferred router. Preferred routers are defined by the order specified in the /ROUTER qualifier of the CREATE FACILITY command. Note that this preference is subject to the router being available and quorate.

/CALL_OUT[=role-list]
/NOCALL_OUT (D)
Specifies which node types are to have call-out servers running on them.

Role-list is a list of roles separated by commas. If role-list contains more than one role then it must be enclosed in parentheses.

role is one of the keywords ROUTER or BACKEND.

The default for role-list is (ROUTER,BACKEND).

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FRONTEND=frontend-list
/NOFRONTEND (D)
Frontend-list is a list of frontend-nodes separated by commas. If there is more than one frontend-node, then frontend-list must be enclosed in parentheses.

Frontend-node is either the name of a node or @file-spec, where file-spec specifies a text file containing a frontend-list on each line.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/RESOURCE_MANAGER=resource-list
Specifies a list of defined resource managers that this facility may reference. Server applications using this facility and resource manager will receive any recovered transactions when the facility is created. See Appendix C for further information.

/REPLY_CHECKSUM
/NOREPLY_CHECKSUM (D)
Specifies that the reply consistency check (or Response Matching) feature for replayed messages is enabled. It is a check for reply consistency during a replay of a reply to client message.

RTR can enable, disable and display this feature.

/ROUTER=router-list
/NOROUTER (D)
Specifies the names of the nodes that act as routers for this facility.

Router-list is a list of router-nodes separated by commas. If there is more than one router-node, then router-list must be enclosed in parentheses.

If /NOBALANCE is specified with the CREATE FACILITY command, then the order in which router nodes are specified with the /ROUTER qualifier defines the preferred routing order.

Router-node is either the name of a node or @file-spec, where file-spec specifies a text file containing a router-list on each line.

Related commands

- DELETE FACILITY
- EXTEND FACILITY
- SHOW FACILITY
- TRIM FACILITY

Examples

See Chapter 2, Starting and Setting Up RTR, for examples of how to use the CREATE FACILITY command.

CREATE JOURNAL

Create RTR's recovery journal.

Format

CREATE JOURNAL [disk-1] ... [,disk-n]

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/BLOCKS=nr-blocks	/BLOCKS=1000
/MAXIMUM_BLOCKS=nr-blocks	/MAXIMUM_BLOCKS=1000
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/SUPERSEDE	/NOSUPERSEDE

Description

The **CREATE JOURNAL** command creates RTR's recovery journal files on the specified disks. The target size of the files is specified using the /BLOCKS qualifier (512 byte blocks). The RTR journaling system will attempt to keep the journal file to this size.

The /MAXIMUM_BLOCKS qualifier specifies the maximum size that the journal file can extend to; the RTR journaling system will not extend the journal file beyond this size.

/BLOCKS and /MAXIMUM_BLOCKS are positional qualifiers, so the journal files need not have the same size on each disk.

The CREATE JOURNAL command checks that a journal does not already exist for the node. An error occurs if a journal does exist, unless the /SUPERSEDE qualifier is specified.

When the /SUPERSEDE qualifier is specified, any previously existing journal files are deleted. For this reason the **CREATE JOURNAL/SUPERSEDE** command should not be issued on a node being started up after a failure if the transactions interrupted by the failure need to be recovered. The CREATE JOURNAL command is normally entered interactively, not automatically from a startup command procedure.

RTR only uses journal files on nodes that are configured to run servers, that is, on backends and on routers with call-out servers.

Parameters

disk-1, ... disk-n
Specifies a list of disk names where the new journal is to reside.

If no disks are specified then the issuer's current default disk will be used.

Spreading the journal over more than one physical disk can improve performance if I/O to the journal file becomes a bottleneck.

Table 6–17 Platform Specific Information

Platform	Journal Root	Finding Disks	Notes
UNIX	/rtrjnl	Use df	Enter disk names as thay appear in /dev. Enclose disk names in quotes and separate names with commas. The journals reside in subdirectories of the /rtrjnl.
OpenVMS	[RTRJNL]	Use SHOW DEVICE	If the SYSTEM account has insufficient disk quota for journal file creation, you must have the EXQUOTA privilege in order for the command to complete successfully.

Qualifiers

/BLOCKS[=nr-blocks]
/BLOCKS=1000 (D)
Specifies the target size of the journal file in 512 byte blocks. This qualifier can be applied locally to each disk or globally for all disks.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/MAXIMUM_BLOCKS[=nr-blocks]
/MAXIMUM_BLOCKS=1000 (D)
Specifies the maximum size that the journal file can use. This qualifier can be applied locally to each disk or globally for all disks.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/SUPERSEDE
/NOSUPERSEDE (D)
Specifies how to handle the case where a journal already exists.

If /SUPERSEDE is specified, then a journal is created whether or not a journal previously existed (unless the previously existing journal is currently in use). The previous contents of the journal, if any, are destroyed.

If /NOSUPERSEDE is specified (default), then a journal is created only if no journal previously existed.

Related commands

- DELETE JOURNAL
- MODIFY JOURNAL
- SHOW JOURNAL

Examples

```
RTR> CREATE JOURNAL /SUPERSEDE DISK1$:/BLOCK=1000/MAX_BLOCK=10000, -
_RTR> DISK2$:/BLOCK=2000/MAX_BLOCK=200000
```

This command deletes any existing journal files and then creates new ones on DISK1$ and DISK2$. The target sizes of the journal files are 1000 and 2000 blocks, and the maximum sizes are 10,000 and 200,000 blocks respectively.

```
RTR> CREATE JOURNAL "/dev/rz3a", "/dev/rz2c" /BLOCK=1000 /MAXIMUM_BLOCK=2000
```

This command creates journal files on /dev/rz3a and /dev/rz2c. The target sizes of the journal files is 1000 blocks and the maximum size of the journal on /dev/rz2c is 2000 blocks.

CREATE PARTITION

Creates an RTR partition.

Format

CREATE PARTITION [partition_name]

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/CONCURRENT	/NOCONCURRENT
/FACILITY=facility-name	
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/STANDBY	/NOSTANDBY
/SHADOW	/NOSHADOW
/KEY1=keysegdesc	
/KEYn=keysegdesc	

Description

The CREATE PARTITION command defines an RTR partition. The partition characteristics that may be defined include key range or ranges and whether attached server processes can be shadows or standbys.

The command must be issued before any server application programs using the partition are started.

Parameters

partition_name
Specifies the name of the partition to be created. Partition names must be unique within a facility.

Any application program which uses this partition must specify the same name when it calls rtr_open_channel( ).

Partition names can contain up to sixty-three (63) characters. Letters, numbers and underline characters are all valid, but the first character of a partition name must be a letter.

If a partition name already exists in the facility, the command fails.

There is no default value for partition_name.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/CONCURRENT (D)
/NOCONCURRENT
Specifies that concurrent servers are allowed for this partition.

/KEYn=keysegdesc
Specifies a partition key segment. Up to nine key segments may be defined for a partition (KEY1, KEY2,... up to KEY9). (If more than nine key segements are required, a named partiton can be created using the rtr_open_channel( ) call.)

The syntax of the KEYn qualifier is:

```
/KEYn=  (type_of_key=[signed|unsigned|string], -
         length_of_key=nnnn, -
         offset_of_key=nnnn, -
         low_bound=[string|nnnn] -
         high_bound=[string|nnnn]
```

type_of_key= Specifies the field type of the key. The key-type must be one of unsigned, signed or string. The default is unsigned.

length_of_key=nnnn Specifies the length of the key field in enqueued messages in bytes. For signed or unsigned ints, length may be either one, two, four or eight bytes. The default value for key-length is four bytes.

offset_of_key=nnnn Specifies the offset of the key within the messages in bytes. The default is zero, that is, the key is at the start of the messages.

low_bound= Specifies the lower bound of the key range that servers in the partition will service. The interpretation of low-bound depends on the key type; if the key is of type string then it is interpreted as text, otherwise it is interpreted as a numeric value. The default for low-bound is the smallest possible value that can fit in the specified key type.

If the key bound value length is less than the key length (given in length_of_key), the key bound will automatically be null-padded to the required length.

high_bound= Specifies the upper bound of the key range that servers in the partition will service. The interpretation of high-bound depends on the key type. If the key is of type string then it is interpreted as text, otherwise it is interpreted as a numeric value. The default for high-bound is the largest possible value that can fit in the specified key type.

If the key bound value length is less than the key length (given in length_of_key), the key bound will automatically be null-padded to the required length.

If the specified key range overlaps that of an existing partition in the facility, or if the key segment description conflicts with an existing definition, the command fails.

/FACILITY
Specifies the name of the facility in which the partition is being created.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/STANDBY (D)
/NOSTANDBY
Specifies that standby servers are allowed for this partition.

/SHADOW
/NOSHADOW (D)
Specifies that shadow servers are allowed for this partition.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file `file-spec`. If `/OUTPUT` or `file-spec` is omitted then the standard or default output is used.

Related Commands

- `CREATE FACILITY`
- `SET PARTITION`

DEFINE /KEY

Assign a string to a keyboard function key.

Format

DEFINE /KEY key-name "equivalence-string"

Command Qualifiers	Defaults
/ECHO	/ECHO
/IF_STATE	/NOIF_STATE
/LOCK_STATE	/NOLOCK_STATE
/LOG	/NOLOG
/SET_STATE	/NOSET_STATE
/TERMINATE	/NOTERMINATE

Description

This command lets you assign a string to a function key, possibly overriding any predefined function that was bound to that key. When you then press the key, the RTR Utility enters the currently associated string into your command line. The RTR DEFINE /KEY command is similar to the OpenVMS DCL DEFINE /KEY command.

A key definition remains in effect until you redefine the key or exit from the RTR Utility. You can include key definitions in command procedures, e.g. in the RTR Utility initialization file.

The /IF_STATE qualifier lets you increase the number of key definitions available on your terminal. The same key can be assigned any number as long as each definition is associated with a different state.

By default, the current key state is the DEFAULT state. The current state may be changed by pressing a key that causes a state change (that is, a key that was defined with the DEFINE /KEY /STATE command).

Parameters

key-name

Specifies a function key to be assigned a string. Table 6–18 describes the valid key names.

Table 6–18 Key names

Key-name	LK201	VT100-type
PF1	PF1	PF1
PF2	PF2	PF2
PF3	PF3	PF3

(continued on next page)

Table 6–18 (Cont.) Key names

Key-name	LK201	VT100-type
PF4	PF4	PF4
KP0, KP1 ..KP9	Keypad 0 .. 9	Keypad 0 .. 9
PERIOD	Keypad period (.)	Keypad period (.)
COMMA	Keypad comma (,)	Keypad comma (,)
MINUS	Keypad minus (-)	Keypad minus (-)
ENTER	ENTER	ENTER
E1	Find	
E2	Insert Here	
E3	Remove	
E4	Select	
E5	Prev Screen	
E6	Next Screen	
HELP	Help	
DO	Do	
F6, F7, .. F20	F6, F7, .. F20	

equivalence-string
Specifies the string to be processed when the specified key is pressed. Typically, this is all or part of an RTR command.

If the string contains spaces or non-alphanumeric characters, it must be enclosed in quotation marks.

Qualifiers

/ECHO (D)
/NOECHO
Controls whether the command line is displayed after the key has been pressed. Do not use /ECHO with /NOTERMINATE.

/IF_STATE=state-name
/NOIF_STATE (D)
Specifies the name of a state to which a key definition applies. /IF_STATE assigns the key definitions to the specified states. A state name may be any appropriate alphanumeric string. /NOIF_STATE (the default) assigns the key definition to the DEFAULT state.

/LOCK_STATE
/NOLOCK_STATE (D)
Controls how long the state set by /SET_STATE remains in effect after the specified key is pressed. /LOCK_STATE causes the state to remain in effect until it is changed explicitly by another function key being pressed that has the /SET_STATE attribute. /NOLOCK_STATE (the default) causes the state to remain in effect only until the next terminator character is typed, or until the next define function key is pressed.

/LOG
/NOLOG (D)
Controls whether a message is displayed indicating that the key definition has been successfully created.

/SET_STATE=state-name
/NOSETSTATE (D)
Controls whether pressing the key changes the current key state. /SET_STATE changes the current state to state-name when you press the key. /NOSET_STATE (the default) causes the current state to remain in effect.

/TERMINATE
/NOTERMINATE (D)
Controls whether the specified string is to be terminated (processed) when the key is pressed. /TERMINATE causes the string to be terminated when the key is pressed. /NOTERMINATE (the default) allows you to press other keys before terminating the string by pressing the RETURN key.

Related Commands

- SHOW KEY

Examples

1.
```
RTR> DEFINE /KEY PF3 "SHOW RTR" /TERMINATE
DEFAULT PF3 key has been defined as "SHOW RTR"
RTR> PF3

RTR running on node BE1
```

The DEFINE /KEY command defines the PF3 key on the keypad to perform a SHOW RTR command. DEFAULT refers to the default state.

2.
```
RTR> DEFINE /KEY PF1 "HELP " /SET_STATE=GOLD /NOTERMINATE /ECHO
DEFAULT PF1 key has been defined as "HELP "
RTR> DEFINE /KEY PF1 " CREATE" /TERMINATE /IF_STATE=GOLD /ECHO
GOLD PF1 key has been defined as "CREATE"
RTR> PF1
RTR> HELP PF1
RTR> HELP CREATE
The "CREATE FACILITY" command is used to ...
```

The first DEFINE /KEY command defines the PF1 key to be the string HELP. The state is set to GOLD for the subsequent key. The /NOTERMINATE qualifier instructs the system remain in command input mode when the key is pressed. The second DEFINE /KEY command defines the use of the PF1 key when the keypad is in the GOLD state. When the keypad is in the GOLD state, pressing PF1 will cause the current read to be terminated.

If you press PF1 twice, the system displays and processes the HELP CREATE command.

The word DEFAULT in the second line of the example refers to the fact that PF1 has been defined in the default state. Note the space before the word CREATE in the second DEFINE /KEY command. If the space is omitted, the system fails to recognize CREATE as the keyword for the HELP command.

DELETE FACILITY

Delete an RTR facility.

Format

DELETE FACILITY facility_name

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The DELETE FACILITY command removes the specified facility on the node where the command is issued. After issuing this command applications are able to use the facility. Any outstanding transactions using the facility are aborted.

Parameters

facility_name
The name of the facility to delete.

The parameter facility_name must be supplied.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- CREATE FACILITY
- SHOW FACILITY

Examples

See Chapter 2, Starting and Setting Up RTR, for examples of how to use the DELETE FACILITY command.

DELETE JOURNAL

Delete an RTR journal.

Format

DELETE JOURNAL

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The DELETE JOURNAL command deletes a previously created RTR journal on the node where the command is issued.

The DELETE JOURNAL command will fail if a journal does not exist, or if a journal has been created but is currently in use. The command causes the previous contents of the journal, if any, to be destroyed.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- CREATE JOURNAL
- MODIFY JOURNAL
- SHOW JOURNAL

Examples

See Chapter 2, Starting and Setting Up RTR, for examples of how to use the DELETE JOURNAL command.

DELETE PARTITION

Delete an RTR PARTITION.

Format

DELETE PARTITION PARTITION_name

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/FACILITY	

Description

The DELETE PARTITION command removes the specified partition on the node where the command is issued.

Parameters

PARTITION_name
The name of the partition to delete.

The parameter partition_name must be supplied. It may be specified as partition_name <or as facility_name:partition_name.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FACILITY
Specifies the name of the facility from which the partition is being deleted.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- CREATE PARTITION
- SHOW PARTITION

DISPLAY BAR

Display a bar-graph in a monitor picture.

Format

DISPLAY BAR expression

Command Qualifiers	Defaults
/AVERAGE=(keyword,...)	/none
/BELL=Boolean-expression	/NOBELL
/BLANK=Boolean-expression	/NOBLANK
/BLINK=Boolean-expression	/NOBLINK
/BOLD=Boolean-expression	/BOLD
/CHARACTER=char	/CHARACTER="a"
/DAMPING=damping-factor	/NODAMPING
/LABEL=text	/NOLABEL
/LENGTH=nr-chars	/LENGTH=50
/MAXIMUM=max-value	/MAXIMUM=10
/MINIMUM=min-value	/MINIMUM=0
/RATE=interval	/NORATE
/REVERSE=Boolean-expression	/NOREVERSE
/ROWS=nr-rows	/ROWS=1
/SELECT=Boolean-expression	/NOSELECT
/SEPARATE=(keyword,...)	/none
/TOTALIZE=(keyword,...)	/none
/UNDERLINE=Boolean-expression	/NOUNDERLINE
/VALUE=value-type	/VALUE=CURRENT
/X=column	Column of previous item
/Y=row	Next free row

Description

The DISPLAY BAR command displays the expression as a bar-graph in a monitor picture. It can be used within a monitor file or issued at the RTR prompt when interactively defining a monitor picture for use in a subsequent MONITOR command.

Note that the introduction of the /SEPARATE, /TOTALIZE and /AVERAGE qualifiers has superseded the qualifiers /FACILITY, /LINK, /PARTITION, /NODE and /PROCESS. These superseded qualifiers are no longer described here, however they are still supported.

Parameters

expression
Specifies the quantity to be displayed. Expression can either be the name of a single data item or an expression combining several items using simple arithmetic operations and constants. In the latter case, EXPRESSION must be in quotes.

Qualifiers

/AVERAGE=(keyword,...)
/NOAVERAGE
Specifies that the items being monitored relating to keyword are displayed as an average. This allows a number of items to be averaged in one qualifier.

The keyword can be one of the following:

Keyword	Meaning
NODE	Node data items
LINK	Link data items
FACILITY	Facility data items
PROCESS	Process data items
PARTITION	Partition data items
FE_TRANSACTION	Frontend transaction data items
TR_TRANSACTION	Backend transaction data items
BE_TRANSACTION	Router transaction data items

/BELL[=Boolean-expression]
/NOBELL (D)
Sends a bell character to the terminal if Boolean-expression evaluates to True (non-zero).

/BLINK[=Boolean-expression]
/NOBLINK (D)
Specifies that the displayed value blinks if Boolean-expression evaluates to True (non-zero).

/BLANK[=Boolean-expression]
/NOBLANK (D)
Specifies that the displayed value is replaced by blanks if Boolean-expression evaluates to True (non-zero).

/BOLD[=Boolean-expression]
/NOBOLD (D)
Specifies that the item is displayed in high intensity if Boolean-expression evaluates to True (non-zero).

/CHARACTER[=char]
/NOCHARACTER
/CHARACTER="a" (D)
Specifies the character used to draw bar charts. The line drawing character set is used to display them. By default, the character "a" is used, which corresponds to a rectangular block in the character set. If /NOCHARACTER is given then no characters are displayed and the bar is visible only if attributes have been specified. This can be used to draw a scale behind the bar chart.

/DAMPING[=damping-factor]
/NODAMPING (D)
Specifies that the value displayed is to fluctuate more slowly than the raw measured value. The default for damping-factor is one. Damping is only relevant if /VALUE=CURRENT.

/LABEL=text
/NOLABEL (D)
Specifies the text used to label the value being displayed. Symbols are substituted at display time. (See Section A.2, Substitution Symbols).

/LENGTH[=nr-chars]
/LENGTH=50 (D)
Specifies the number of characters in a bar chart representing the maximum value. The default is 50.

/MAXIMUM[=max-value]
/MAXIMUM=10 (D)
Specifies the maximum value represented on a bar chart. The default is 10.

/MINIMUM[=min-value]
/MINIMUM=0 (D)
Specifies the minimum value represented on a bar chart. The default is zero.

/RATE[=interval]
/NORATE (D)
Specifies that the rate of change of the expression is to be displayed rather than the absolute value. When /RATE is used, interval specifies the time interval in seconds used to calculate the rate of change. This has no effect on the sampling, it simply allows the rate to be displayed in another unit. For example, displaying the start transaction counter with /RATE=60 results in transactions per minute being displayed instead of per second.

/REVERSE[=Boolean-expression]
/NOREVERSE (D)
Specifies that the item is displayed with the foreground and background visual attributes swapped if Boolean-expression evaluates to True (non-zero).

/ROWS[=nr-rows]
/ROWS=1 (D)
Specifies how many rows are used to display the item. This is only meaningful if /SEPARATE is also specified. The default number of rows is one.

/SELECT[=Boolean-expression]
/NOSELECT (D)
Displays the item if Boolean-expression evaluates to True (non-zero).

/SEPARATE=(keyword,...)
/NOSEPARATE
Specifies that the items being monitored relating to keyword are separated from each other and displayed as a list. This allows a number of items to be separated in one qualifier.

The keyword can be one of the following:

Keyword	Meaning
NODE	Node data items
LINK	Link data items
FACILITY	Facility data items

Keyword	Meaning
PROCESS	Process data items
PARTITION	Partition data items
FE_TRANSACTION	Frontend transaction data items
TR_TRANSACTION	Backend transaction data items
BE_TRANSACTION	Router transaction data items

/TOTALIZE=(keyword,...)
/NOTOTALIZE
Specifies that the items being monitored relating to keyword are added together and displayed as a total.

The keyword can be one of the following:

Keyword	Meaning
NODE	Node data items
LINK	Link data items
FACILITY	Facility data items
PROCESS	Process data items
PARTITION	Partition data items
FE_TRANSACTION	Frontend transaction data items
TR_TRANSACTION	Backend transaction data items
BE_TRANSACTION	Router transaction data items

/UNDERLINE[=Boolean-expression]
/NOUNDERLINE (D)
Specifies that the displayed value is underlined if Boolean-expression evaluates to True (non-zero).

/VALUE[=value-type]
/VALUE=CURRENT (D)
Specifies how the value is processed before being displayed. Value-type can be one of the following keywords:

- CURRENT - (default) display the current value of expression
- AVERAGE - display the average value of expression since the MONITOR command was issued.
- MINIMUM - display the minimum value of expression since the MONITOR command was issued.
- MAXIMUM - display the maximum value of expression since the MONITOR command was issued.

Use the MONITOR/RESUME command to reset average, maximum or minimum values.

/X[=column]

/X=previous-column (D)
Specifies the screen column where the item is displayed (the leftmost column is 1). By default, items are displayed in the same column as defined by the previous DISPLAY command.

/Y[=row]
/Y=next-free-row (D)
Specifies the screen row where the item is displayed (top row is 1). By default, items are displayed on the next free row after the item defined by the previous DISPLAY command.

Related Commands

- MONITOR
- SHOW DISPLAY
- CLEAR
- DISPLAY NUMERIC
- DISPLAY SYMBOLIC
- DISPLAY TEXT

Examples

See Section A.1 for examples of how to use the DISPLAY BAR command.

DISPLAY NUMERIC

Display a number in a monitor picture.

Format

DISPLAY NUMERIC expression

Command Qualifiers	Defaults
/AVERAGE=(keyword,...)	/none
/BELL=Boolean-expression	/NOBELL
/BLANK=Boolean-expression	/NOBLANK
/BLINK=Boolean-expression	/NOBLINK
/BOLD=Boolean-expression	/NOBOLD
/DAMPING=damping-factor	/NODAMPING
/DECIMALS=decimal-places	/DECIMALS=0
/LABEL=text	/NOLABEL
/RATE=interval	/NORATE
/REVERSE=Boolean-expression	/NOREVERSE
/ROWS=nr-rows	/ROWS=1
/SELECT=Boolean-expression	/NOSELECT
/SEPARATE=(keyword,...)	/none
/TOTALIZE=(keyword,...)	/none
/UNDERLINE=Boolean-expression	/NOUNDERLINE
/VALUE=value-type	/VALUE=CURRENT
/WIDTH=field-width	/WIDTH=1
/X=column	Column of previous item
/Y=row	Next free row

Description

The DISPLAY NUMERIC command displays the specified expression as a number in a monitor picture. It can be used within a monitor file or issued at the RTR prompt when interactively defining a monitor picture for use in a subsequent MONITOR command.

Note that the introduction of the /SEPARATE, /TOTALIZE and /AVERAGE qualifiers has superseded the qualifiers /FACILITY, /LINK, /PARTITION, /NODE and /PROCESS. These superseded qualifiers are no longer described here, however they are still supported.

Parameters

expression
Specifies the quantity to be displayed. Expression can either be the name of a single data item or an expression combining several items using simple arithmetic operations and constants. In the latter case, EXPRESSION must be in quotes.

Qualifiers

/AVERAGE=(keyword,...)
/NOAVERAGE
Specifies that the items being monitored relating to keyword are displayed as an average. This allows a number of items to be averaged in one qualifier.

The keyword can be one of the following:

Keyword	Meaning
NODE	Node data items
LINK	Link data items
FACILITY	Facility data items
PROCESS	Process data items
PARTITION	Partition data items
FE_TRANSACTION	Frontend transaction data items
TR_TRANSACTION	Backend transaction data items
BE_TRANSACTION	Router transaction data items

/BELL[=Boolean-expression]
/NOBELL (D)
Sends a bell character to the terminal if Boolean-expression evaluates to True (non-zero).

/BLANK[=Boolean-expression]
/NOBLANK (D)
Specifies that the displayed value is replaced by blanks if Boolean-expression evaluates to True (non-zero).

/BLINK[=Boolean-expression]
/NOBLINK (D)
Specifies that the displayed value blinks if Boolean-expression evaluates to True (non-zero).

/BOLD[=Boolean-expression]
/NOBOLD (D)
Specifies that the item is displayed in high intensity if Boolean-expression evaluates to True (non-zero).

/DAMPING[=damping-factor]
/NODAMPING (D)
Specifies that the value displayed is to fluctuate more slowly than the raw measured value. The default for damping-factor is one. Damping is only relevant if /VALUE=CURRENT.

/DECIMALS[=decimal-places]
/DECIMALS=0 (D)
Specifies the number of digits to appear after the decimal point when displaying a numeric. The default for decimal-places is zero, that is, numbers are displayed as integers.

/LABEL=text
/NOLABEL (D)
Specifies the text used to label the value being displayed. Symbols are substituted at display time. (See Section A.2, Substitution Symbols).

/RATE[=interval]
/NORATE (D)
Specifies that the rate of change of the expression is to be displayed rather than the absolute value. When /RATE is used, interval specifies the time interval in seconds used to calculate the rate of change. This has no effect on the sampling, it simply allows the rate to be displayed in another unit. For example, displaying the start transaction counter with /RATE=60 results in transactions per minute being displayed instead of per second.

/REVERSE[=Boolean-expression]
/NOREVERSE (D)
Specifies that the item is displayed with the foreground and background visual attributes swapped if Boolean-expression evaluates to True (non-zero).

/ROWS[=nr-rows]
/ROWS=1 (D)
Specifies how many rows are used to display the item. This is only meaningful if /SEPARATE is also specified. The default number of rows is one.

/SELECT[=Boolean-expression]
/NOSELECT (D)
Displays the item if Boolean-expression evaluates to True (non-zero).

/SEPARATE=(keyword,...)
/NOSEPARATE
Specifies that the items being monitored relating to keyword are separated from each other and displayed as a list. This allows a number of items to be separated in one qualifier.

The keyword can be one of the following:

Keyword	Meaning
NODE	Node data items
LINK	Link data items
FACILITY	Facility data items
PROCESS	Process data items
PARTITION	Partition data items
FE_TRANSACTION	Frontend transaction data items
TR_TRANSACTION	Backend transaction data items
BE_TRANSACTION	Router transaction data items

/TOTALIZE=(keyword,...)
/NOTOTALIZE
Specifies that the items being monitored relating to keyword are added together and displayed as a total.

The keyword can be one of the following:

Keyword	Meaning
NODE	Node data items

Keyword	Meaning
LINK	Link data items
FACILITY	Facility data items
PROCESS	Process data items
PARTITION	Partition data items
FE_TRANSACTION	Frontend transaction data items
TR_TRANSACTION	Backend transaction data items
BE_TRANSACTION	Router transaction data items

/UNDERLINE[=Boolean-expression]
/NOUNDERLINE (D)
Specifies that the displayed value is underlined if Boolean-expression evaluates to True (non-zero).

/VALUE[=value-type]
/VALUE=CURRENT (D)
Specifies how the value is processed before being displayed. Value-type can be one of the following keywords:

- CURRENT - (default) display the current value of expression
- AVERAGE - display the average value of expression since the MONITOR command was issued.
- MINIMUM - display the minimum value of expression since the MONITOR command was issued.
- MAXIMUM - display the maximum value of expression since the MONITOR command was issued.

Use the MONITOR/RESUME command to reset average, maximum or minimum values.

/WIDTH[=field-width]
/WIDTH=1 (D)
Specifies the width (in number of characters) to display a numeric. If more characters than width are required then the number will be shifted to the right. No data will be lost but columns of numbers will no longer line up. If /WIDTH=1 is specified (the default) then numbers are displayed left justified.

/X[=column]
/X=previous-column (D)
Specifies the screen column where the item is displayed (the leftmost column is 1). By default, items are displayed in the same column as defined by the previous DISPLAY command.

/Y[=row]
/Y=next-free-row (D)
Specifies the screen row where the item is displayed (top row is 1). By default, items are displayed on the next free row after the item defined by the previous DISPLAY command.

Related Commands

- MONITOR
- SHOW DISPLAY
- CLEAR
- DISPLAY BAR
- DISPLAY SYMBOLIC
- DISPLAY TEXT

Examples

See Section A.1 for examples of how to use the DISPLAY NUMERIC command.

DISPLAY STRING

Display a string in a monitor picture.

Format

DISPLAY STRING expression

Command Qualifiers	Defaults
/AVERAGE=(keyword,...)	/none
/BELL=Boolean-expression	/NOBELL
/BLANK=Boolean-expression	/NOBLANK
/BLINK=Boolean-expression	/NOBLINK
/BOLD=Boolean-expression	/BOLD
/JUSTIFY=keyword	/JUSTIFY=LEFT
/LABEL=text	/NOLABEL
/REVERSE=Boolean-expression	/NOREVERSE
/ROWS=nr-rows	/ROWS=1
/SELECT=Boolean-expression	/NOSELECT
/SEPARATE=(keyword,...)	/none
/TOTALIZE=(keyword,...)	/none
/UNDERLINE=Boolean-expression	/NOUNDERLINE
/WIDTH=field-width	/WIDTH=0
/X=column	Column of previous item
/Y=row	Next free row

Description

The DISPLAY STRING command displays the specified expression as an alphanumeric in a monitor picture. It can be used within a monitor file or issued at the RTR prompt when interactively defining a monitor picture for use in a subsequent MONITOR command.

Note that the introduction of the /SEPARATE, /TOTALIZE and /AVERAGE qualifiers has superseded the qualifiers /FACILITY, /LINK, /PARTITION, /NODE and /PROCESS. These superseded qualifiers are no longer described here, however they are still supported.

Parameters

expression
The string you want to display; enclose it in quote marks. Expression can be the name of a single RTR data item.

Qualifiers

/AVERAGE=(keyword,...)
/NOAVERAGE
Specifies that the items being monitored relating to keyword are displayed as an average. This allows a number of items to be averaged in one qualifier.

The keyword can be one of the following:

Keyword	Meaning
NODE	Node data items
LINK	Link data items
FACILITY	Facility data items
PROCESS	Process data items
PARTITION	Partition data items
FE_TRANSACTION	Frontend transaction data items
TR_TRANSACTION	Backend transaction data items
BE_TRANSACTION	Router transaction data items

/BLANK[=Boolean-expression]
/NOBLANK (D)
Specifies that the displayed value is replaced by blanks if Boolean-expression evaluates to True (non-zero).

/BLINK[=Boolean-expression]
/NOBLINK (D)
Specifies that the displayed value blinks if Boolean-expression evaluates to True (non-zero).

/BOLD[=Boolean-expression]
/NOBOLD (D)
Specifies that the item is displayed in high intensity if Boolean-expression evaluates to True (non-zero).

/JUSTIFY=keyword
/JUSTIFY=LEFT (D)
Specifies whether justification is left, right or centered in the available width. (The keyword may be LEFT, RIGHT or CENTERED.) The /WIDTH qualifier must be greater than zero for the /JUSTIFY to operate. If the available width is smaller than the string, truncation will occur.

/LABEL=text
/NOLABEL (D)
Specifies the text used to label the value being displayed. Symbols are substituted at display time. (See Section A.2, Substitution Symbols).

/REVERSE[=Boolean-expression]
/NOREVERSE (D)
Specifies that the item is displayed with the foreground and background visual attributes swapped if Boolean-expression evaluates to True (non-zero).

/ROWS[=nr-rows]
/ROWS=1 (D)
Specifies how many rows are used to display the item. This is only meaningful if /SEPARATE is also specified. The default number of rows is one.

/SELECT[=Boolean-expression]
/NOSELECT (D)
Displays the item if Boolean-expression evaluates to True (non-zero).

/SEPARATE=(keyword,...)
/NOSEPARATE
Specifies that the items being monitored relating to keyword are separated from each other and displayed as a list. This allows a number of items to be separated in one qualifier.

The keyword can be one of the following:

Keyword	Meaning
NODE	Node data items
LINK	Link data items
FACILITY	Facility data items
PROCESS	Process data items
PARTITION	Partition data items
FE_TRANSACTION	Frontend transaction data items
TR_TRANSACTION	Backend transaction data items
BE_TRANSACTION	Router transaction data items

/TOTALIZE=(keyword,...)
/NOTOTALIZE
Specifies that the items being monitored relating to keyword are added together and displayed as a total.

The keyword can be one of the following:

Keyword	Meaning
NODE	Node data items
LINK	Link data items
FACILITY	Facility data items
PROCESS	Process data items
PARTITION	Partition data items
FE_TRANSACTION	Frontend transaction data items
TR_TRANSACTION	Backend transaction data items
BE_TRANSACTION	Router transaction data items

/UNDERLINE[=Boolean-expression]
/NOUNDERLINE (D)
Specifies that the displayed value is underlined if Boolean-expression evaluates to True (non-zero).

/WIDTH[=field-width]
/WIDTH=0 (D)
Specifies the width (in number of characters) to display the string. If /WIDTH=0 is specified (the default) then the string is displayed as entered. If the width is greater than zero, then the /JUSTIFY qualifier is activated.

/X[=column]
/X=previous-column (D)
Specifies the screen column where the item is displayed (the leftmost column is 1). By default, items are displayed in the same column as defined by the previous DISPLAY command.

/Y[=row]
/Y=next-free-row (D)
Specifies the screen row where the item is displayed (top row is 1). By default, items are displayed on the next free row after the item defined by the previous DISPLAY command.

Related Commands

- MONITOR
- SHOW DISPLAY
- CLEAR
- DISPLAY BAR
- DISPLAY SYMBOLIC
- DISPLAY TEXT

Examples

See Section A.1 for examples of how to use the DISPLAY STRING command.

DISPLAY SYMBOLIC

Display a text in a monitor picture depending on the result of an expression evaluation.

Format

DISPLAY SYMBOLIC expression "text-string" [,"text-string"]...

Command Qualifiers	Defaults
/BELL=Boolean-expression	/NOBELL
/BLANK=Boolean-expression	/NOBLANK
/BLINK=Boolean-expression	/NOBLINK
/BOLD=Boolean-expression	/BOLD
/REVERSE	/NOREVERSE
/UNDERLINE=Boolean-expression	/NOUNDERLINE
/X=column	Column of previous item
/Y=row	Next free row

Description

The DISPLAY SYMBOLIC command displays one of the text strings depending on the value of expression. The first string is output if the expression's value is zero (0), the second string is output if the expression's value is 1, and so on. If the expression has a value for which there is no corresponding entry in the list of texts, then the value itself is printed. (Note that there is a limit of 255 characters on the size of one command to RTR, so large numbers of long strings should be avoided.)

The command can be used within a monitor file or issued at the RTR prompt when interactively defining a monitor picture for use in a subsequent MONITOR command.

Parameters

expression
The expression to be evaluated. Expression can either be the name of a single data item, or an expression combining several data items using simple arithmetic operations and constants. In the latter case, the data items must all be of the same type and EXPRESSION must be enclosed in quotation marks.

Qualifiers

/BLANK[=Boolean-expression]
/NOBLANK (D)
Specifies that the displayed value is replaced by blanks if Boolean-expression evaluates to True (non-zero).

/BLINK[=Boolean-expression]
/NOBLINK (D)
Specifies that the displayed value blinks if Boolean-expression evaluates to True (non-zero).

/BOLD[=Boolean-expression]
/NOBOLD (D)
Specifies that the item is displayed in high intensity if Boolean-expression evaluates to True (non-zero).

/REVERSE[=Boolean-expression]
/NOREVERSE (D)
Specifies that the item is displayed with the foreground and background visual attributes swapped if Boolean-expression evaluates to True (non-zero).

/UNDERLINE[=Boolean-expression]
/NOUNDERLINE (D)
Specifies that the displayed value is underlined if Boolean-expression evaluates to True (non-zero).

/X[=column]
/X=previous-column (D)
Specifies the screen column where the item is displayed (the leftmost column is 1). By default, items are displayed in the same column as defined by the previous DISPLAY command.

/Y[=row]
/Y=next-free-row (D)
Specifies the screen row where the item is displayed (top row is 1). By default, items are displayed on the next free row after the item defined by the previous DISPLAY command.

Related Commands

- MONITOR
- SHOW DISPLAY
- CLEAR
- DISPLAY BAR
- DISPLAY NUMERIC
- DISPLAY TEXT

Examples

See Section A.1 for examples of how to use the DISPLAY SYMBOLIC command.

DISPLAY TEXT

Display text in a monitor picture.

Format

DISPLAY TEXT text

Command Qualifiers	Defaults
/BELL=Boolean-expression	/NOBELL
/BLANK=Boolean-expression	/NOBLANK
/BLINK=Boolean-expression	/NOBLINK
/BOLD=Boolean-expression	/NOBOLD
/FACILITY	/NOFACILITY
/LINK	/NOLINK
/NODE	/NODE
/PROCESS	/NOPROCESS
/REVERSE=Boolean-expression	/NOREVERSE
/SELECT=Boolean-expression	/NOSELECT
/UNDERLINE	/NOUNDERLINE
/X=column	Column of previous item
/Y=row	Next free row

Description

The DISPLAY TEXT displays the specified text in a monitor picture. It can be used within a monitor file or issued at the RTR prompt when interactively defining a monitor picture for use in a subsequent MONITOR command.

Parameters

text
Specifies the text to be displayed. This text may contain any of the substitution symbols. See Section A.2, Substitution Symbols.

Qualifiers

/BELL[=Boolean-expression]
/NOBELL (D)
Sends a bell character to the terminal if Boolean-expression evaluates to True (non-zero).

/BLANK[=Boolean-expression]
/NOBLANK (D)
Specifies that the displayed value is replaced by blanks if Boolean-expression evaluates to True (non-zero).

/BLINK[=Boolean-expression]
/NOBLINK (D)
Specifies that the displayed value blinks if Boolean-expression evaluates to True (non-zero).

/BOLD[=Boolean-expression]
/NOBOLD (D)
Specifies that the item is displayed in high intensity if Boolean-expression evaluates to True (non-zero).

/FACILITY
/NOFACILITY (D)
Specifies that the symbol substitution in the text is carried out as if a facility data item were being displayed. This means that the link name symbol ($LINK_NAME) and the process related symbols ($PROCESS_ID, $PROCESS_NAME, $IMAGE_NAME, $FULL_IMAGE_NAME) are always replaced by the text "-ALL-".

The facility name symbol ($FACILITY_NAME) will be replaced by the text "-ALL-" unless MONITOR/FACILITY=facility-name is used; in this case $FACILITY_NAME is replaced by facility-name.

/LINK
/NOLINK (D)
Specifies that symbol substitution in the text is carried out as if a link data item were being displayed. This means that the facility name symbol ($FACILITY_NAME) and the process related symbols ($PROCESS_ID, $PROCESS_NAME, $IMAGE_NAME, $FULL_IMAGE_NAME) are always replaced by the text "-ALL-". The link name symbol ($LINK_NAME) is replaced by the text "-ALL-" unless MONITOR/LINK=node-name is used, in which case $LINK_NAME is replaced by node-name.

/NODE
/NONODE
Specifies that symbol substitution in the text is carried out as if a node data item were being displayed. This means that the facility name symbol ($FACILITY_NAME), the link name symbol ($LINK_NAME) and the process related symbols ($PROCESS_ID, $PROCESS_NAME, $IMAGE_NAME, $FULL_IMAGE_NAME) are always replaced by the text "-ALL-".

/PROCESS
/NOPROCESS (D)
Specifies that symbol substitution in the text is carried out as if a process data item were being displayed. This means that the facility name symbol ($FACILITY_NAME) and the link name symbol ($LINK_NAME) are always replaced by the text "-ALL-".

The process related symbols ($PROCESS_ID, $PROCESS_NAME, $IMAGE_NAME, $FULL_IMAGE_NAME) are replaced by the text "-ALL-" unless MONITOR /IDENTIFICATION=process-id is used. In this case they are replaced by the appropriate strings for the process specified by process-id.

/REVERSE[=Boolean-expression]
/NOREVERSE (D)
Specifies that the item is displayed with the foreground and background visual attributes swapped if Boolean-expression evaluates to True (non-zero).

/SELECT[=Boolean-expression]
/NOSELECT (D)
Displays the item if Boolean-expression evaluates to True (non-zero).

/UNDERLINE[=Boolean-expression]
/NOUNDERLINE (D)
Specifies that the displayed value is underlined if Boolean-expression evaluates to True (non-zero).

/X[=column]
/X=previous-column (D)
Specifies the screen column where the item is displayed (the leftmost column is 1). By default, items are displayed in the same column as defined by the previous DISPLAY command.

/Y[=row]
/Y=next-free-row (D)
Specifies the screen row where the item is displayed (top row is 1). By default, items are displayed on the next free row after the item defined by the previous DISPLAY command.

Related Commands

- MONITOR
- SHOW DISPLAY
- CLEAR
- DISPLAY NUMERIC
- DISPLAY BAR
- DISPLAY SYMBOLIC

Examples

See Section A.1, Interactive Definition of a Monitor Picture for examples of how to use the DISPLAY TEXT command.

DO

Execute an operating system command.

Format

DO [operating-system-command]

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The DO command enables an operating system command to be executed from RTR. By using the /NODE and /CLUSTER qualifiers the command can be executed on one or more remote nodes. (Note that the SPAWN command does not have this ability).

The DO command is only suitable for commands producing line oriented output, use SPAWN if you want to execute a local operating system command that produces screen oriented output (for example, the OpenVMS MONITOR SYSTEM command, or screen mode editors).

Parameters

DCL-command
The operating system command that you want to execute.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- SPAWN

Example

```
RTR> DO/CLUSTER SHOW TIME
```

This command shows the time on all nodes in a OpenVMS cluster.

```
RTR> DO/NODE=(TR2,TR1) SHOW LOGICAL MYLOGICAL
```

This command examines the logical name "MYLOGICAL" on nodes TR2 and TR1.

```
RTR> SET ENVIRONMENT/NODE=(TR2,TR1)
RTR> DO SHOW TIME
RTR> DO SHOW LOGICAL MYLOGICAL
```

The SET ENVIRONMENT command can be used if a series of DCL commands are to be issued on the same nodes.

Example

```
RTR> DO/NODE=(TR2,TR1) "ps"
```

This command displays the processes running on Compaq Tru64 UNIX nodes TR2 and TR1.

FLUSH NAME_CACHE

Flushes RTR's internal network name cache.

Format

FLUSH NAME_CACHE

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE[=node-list]	/NODE=default-node-list

Description

The FLUSH NAME_CACHE removes information for all known nodes from RTR's internal network name cache.

Network links could become unstable if a Distributed Name Service (DNS) was configured improperly or the service was slow in responding. During extreme DNS latency, RTR could time-out the connections to nodes waiting for a DNS response. To avoid these problems, RTR has implemented an internal node-name-to-id cache; this reduces RTR's exposure to degraded name servers. The contents of the cache can be deleted the command FLUSH NAME_CACHE.

FLUSH NAME_CACHE may be used if the network has been reconfigured or nodes have changed their addresses.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

EXECUTE

Executes a file containing RTR commands.

Format

EXECUTE file-spec

Command Qualifiers	Defaults
/VERIFY	/NOVERIFY

Description

The **EXECUTE** command reads a file containing RTR commands and executes them. This command also has the form `@file-spec`.

Parameters

file-spec
Specifies the name of the file containing commands to be executed.

Qualifiers

/VERIFY
/NOVERIFY (D)
Specifies that the commands being executed and the resulting information is displayed on the terminal.

Examples

```
RTR> execute facility_startup
```

This command executes the file `facility_startup`. This file might contain commands such as:

```
start rtr
create journal
create facility funding/fontend=(node1,node2)/router=(node3)...
```

EXIT

Exits from the RTR prompt.

Format

EXIT

Description

The **EXIT** command exits from the RTR prompt and returns control to the operating system prompt. The command has no parameters or qualifiers. Same as **QUIT**.

EXTEND FACILITY

Adds new nodes or roles or both to an existing facility definition.

Format

EXTEND FACILITY [facility_name]

Command Qualifiers	Defaults
/BACKEND=backend-list	/NOBACKEND
/BALANCE	/NOBALANCE
/CALL_OUT=role-list	/NOCALL_OUT
/CLUSTER	/NOCLUSTER
/FRONTEND=frontend-list	/NOFRONTEND
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/ROUTER=router-list	/NOROUTER

Description

The EXTEND FACILITY command extends the configuration of an RTR facility. New nodes and roles can be added to a facility definition using the EXTEND FACILITY command. Thus a new node can be introduced into a facility, or a new role can be added to an existing node (for example, a router node can be extended to have a backend role).

Notes:

New router nodes must have all backend nodes defined and new backend nodes must have all router nodes defined.

Routers need only be defined with the frontends that they can connect to.

Frontends need only be defined with the routers they can connect to.

When using this command, the facility being extended may temporarily lose quorum until the affected nodes agree upon the new facility definition. During this time server applications will not be presented with any new transactions.

The RTR MONITOR QUORUM displays a monitor picture which allows the quorum negotiations to be followed after a TRIM or EXTEND of a facility. Once quorum has been attained, the participating nodes return to state "qrt".

As with CREATE FACILITY, nodes or roles may be specified which are superfluous. That is, you may specify backend nodes on a node which only has a frontend role, and frontend nodes may be specified on a node which only has a backend role. This permits a single RTR management command to be issued on many nodes, and each node only accepts those parts of the command which are relevant to it.

For example, in a two node facility called facnam, the node FE has the frontend role only, and node FETRBE that has frontend, router and backend roles can be created as follows:

```
$ RTR
RTR> SET ENVIRONMENT /NODE=(FE,FETRBE)
RTR> CREATE FACILITY facnam /FRONTEND=(FE,FETRBE) -
    /ROUTER=FETRBE -
    /BACKEND=FETRBE
```

A new frontend NFE can be added to this facility as follows:

```
$ RTR
RTR> SET ENVIRONMENT /NODE=(FETRBE,NFE)
RTR> EXTEND FACILITY facnam /FRONTEND=NFE -
    /ROUTER=FETRBE
```

Parameters

facility_name

Specifies the name of the facility to be extended.

Any application program which uses this facility must specify the same name when it calls the rtr_open_channel.

Facility names can contain up to thirty-one characters. Letters, numbers and underline characters are all valid, but the first character of a facility name must be a letter.

The default value for facility_name is RTR$DEFAULT_FACILITY.

The /ROUTER qualifier, and at least one of /FRONTEND or /BACKEND must be specified.

An EXTEND FACILITY command executed on a node where the facility is not defined is interpreted as a CREATE FACILITY command.

To maintain consistency in a facility, the following rules apply when adding nodes or roles or both:

- If a frontend role is added, then the node where it is added must know about at least one router of the facility.
- If a backend role is added, then the node where it is added must know about all routers of the facility. If any router nodes are not known by a backend, then problems may occur in achieving quorum.
- If a router role is added, then the node upon which it is added must know about all backends of the facility. If not all backend nodes are known by a router, then problems will occur in achieving quorum. At least one frontend must be defined on a node that has the router role.
- In order to have a consistent facility definition across all nodes in the facility (and thus avoid problems in attaining quorum), the command to add a router role must be executed on all relevant nodes, that is, the node gaining the router role and all backend nodes.

Qualifiers

/BACKEND=backend-list
/NOBACKEND (D)

Specifies the names of the added nodes that are to act as backends for this facility.

Backend-list is a list of backend-nodes separated by commas. If there is more than one backend-node, then backend-list must be enclosed in parentheses.

Backend-node is either the name of a node or @file-spec, where file-spec specifies a file containing a backend-list on each line.

/BALANCE
/NOBALANCE (D)
Specifies that load balancing is enabled for frontend/router connections across the facility.

In order for load balancing to function correctly, /BALANCE must be defined on all routers, as well as on those frontends requiring load balancing.

It has no significance on a backend node, and will be ignored if specified.

The default behavior (/NOBALANCE) is for a frontend to connect to the preferred Router. Preferred routers are defined by the order specified in the /ROUTER qualifier of the EXTEND FACILITY command. Note that this preference is subject to the router being available and quorate.

For more details on frontend load balancing, see Section 2.6, Router Load Balancing.

/CALL_OUT[=role-list]
/NOCALL_OUT (D)
Specifies which node types are to have call-out servers running on them.

Role-list is a comma separated list of roles. If role-list contains more than one role then it must be enclosed in parentheses.

Role is one of the keywords ROUTER or BACKEND.

The default for role-list is (ROUTER,BACKEND).

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FRONTEND=frontend-list
/NOFRONTEND (D)
Specifies the names of the added nodes that act as frontends in this facility. Frontend-list is a list of frontend-nodes separated by commas. If there is more than one frontend-node, then frontend-list must be enclosed in parentheses.

Frontend-node is either the name of a node or @file-spec, where file-spec specifies a text file containing a frontend-list on each line.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/ROUTER=router-list
/NOROUTER (D)
Specifies the names of the added nodes that act as routers for this facility.

Router-list is a list of router-nodes separated by commas. If there is more than one router-node, then router-list must be enclosed in parentheses.

If /NOBALANCE is specified with the EXTEND FACILITY command, then the order in which router nodes are specified with the /ROUTER qualifier defines the preferred routing order.

Router-node is either the name of a node or @file-spec.

File-spec specifies a text file containing a router-list on each line.

Related Commands

- CREATE FACILITY
- DELETE FACILITY
- SHOW FACILITY
- TRIM FACILITY

Examples

See Chapter 2, Starting and Setting Up RTR, for examples of how to use the EXTEND FACILITY command.

INITIALIZE JOURNAL

See CREATE JOURNAL; INITIALIZE is only retained for compatibility reasons.

LOG

Specify RTR to write a log message to a log file.

Format

LOG

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The LOG command specifies a defined log entry to be written to its log messages. You can write log messages to the operator console and to a maximum of four log files. Log files must be periodically purged to avoid difficulties with full disks. Do this by using SET LOG to specify a new file and deleting the old one.

If neither the /OPERATOR nor the /FILE qualifier is specified then logging is suppressed.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- SHOW LOG

Examples

```
RTR> LOG/OUTPUT=RTRLOG.LOG "Message check here"
```

This command tells RTR to write a log message to the file RTRLOG.LOG.

```
RTR> LOG/CLUSTER="Check for this message to see if
logging is working"
```

This command tells RTR to write log messages to all members of a cluster.

```
RTR> LOG/NODE=hostname "Message check HERE"
```

This command tells RTR to write defined log message to the log to the hostname on the node list.

MODIFY JOURNAL

Specifies the desired and maximum allowed sizes of RTR's recovery journal.

Format

MODIFY JOURNAL [disk-1] ... [,disk-n]

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/BLOCKS=nr-blocks	/BLOCKS=1000
/MAXIMUM_BLOCKS=nr-blocks	/MAXIMUM_BLOCKS=1000
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The **MODIFY JOURNAL** command specifies how the size of RTR recovery journal files on the specified disks can be modified. The target or desired size is specified using the /BLOCKS qualifier. The maximum_allowed size is specified using the /MAXIMUM_BLOCKS qualifier. /BLOCKS and /MAXIMUM_BLOCKS are a positional qualifiers, so journal files need not be the same size on each disk.

RTR only uses journal files on nodes that are configured to run servers, that is, on backends and on routers with call-out servers.

Note that the MODIFY JOURNAL command does not cause immediate journal file extension. Actual file size modifications take place on demand (by the RTRACP) within the limits defined by the MODIFY JOURNAL command.

The MODIFY JOURNAL command assumes that a journal already exists for the node. If a journal does not exist, an error message is output.

In contrast to the CREATE JOURNAL command, the MODIFY JOURNAL command is normally entered interactively, not automatically from a startup command procedure.

Parameters

disk-1 ... disk-n
Specifies a list of disk names where journal files are modified.

Refer to the CREATE JOURNAL command for information about disks used for journal files.

Qualifiers

/BLOCKS[=nr-blocks]
/BLOCKS=1000 (D)
Specifies the size of the journal file in blocks. This qualifier can be applied locally to each disk or globally for all disks.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/MAXIMUM_BLOCKS[=nr-blocks]
/MAXIMUM_BLOCKS=1000 (D)
Specifies the maximum size that the journal file can use. This qualifier can be applied locally to each disk or globally for all disks.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- CREATE JOURNAL
- INITIALIZE JOURNAL
- SHOW JOURNAL

OpenVMS Example

```
RTR> MODIFY JOURNAL DISK1$:/BLOCKS=3000/MAXIMUM_BLOCKS=20000
```

This command specifies that the desired size of the journal file is 3000 blocks, and the maximum journal file size is 20,000 blocks.

Example

```
RTR> MODIFY JOURNAL "/dev/rz3a" /BLOCK=2000 /MAXIMUM_BLOCKS=20000
```

This command specifies that the desired size of the journal file is 2000 blocks, and the maximum journal file size is 20,000 blocks.

MONITOR

Displays a monitor picture on the screen.

Format

MONITOR [monitor-file-spec]

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/COUNT=nr-updates	/COUNT=infinite
/FACILITY=facility-name	/NOFACILITY
/IDENTIFICATION=process-id	/NOIDENTIFICATION
/INTERVAL=delay-seconds	/INTERVAL=2
/LINK=link-name	/NOLINK
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/PARTITION=partition-name	/NOPARTITION
/RESUME	/NORESUME
/VERIFY	/NOVERIFY

Description

The MONITOR command allows certain RTR status variables to be continuously displayed on your terminal.

The individual items displayed in the monitor picture may be defined interactively using DISPLAY commands and then executed using a MONITOR /RESUME command.

You may also put the DISPLAY commands into a file (called a monitor file) and then issue a MONITOR monitor-file-spec command.

Parameters

monitor-file-spec
Specifies a file containing DISPLAY commands. Monitor file names are of the form monitor-file-spec.mon

This file may specify either a user defined display or one of the standard displays supplied with RTR. If monitor-file-spec contains only the file-name portion of a file specification then the RTR utility first searches the platform-specific location(s) for a standard monitor file.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that all nodes in the cluster are monitored.

If neither /NODE nor /CLUSTER is specified then the nodes specified with the latest SET ENVIRONMENT command is monitored. If no SET ENVIRONMENT has been issued then the MONITOR command is executed on the node where it was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/COUNT=nr-updates
/COUNT=infinite (D)
Specifies how many times the RTR utility updates the screen before exiting or returning to the RTR> prompt.

The default is that RTR updates the screen until CTRL-Z, CTRL-Y or another RTR command is entered. The /COUNT qualifier may be used when the /OUTPUT qualifier is being used to redirect output to a file. In this case, nr-updates specifies how many screen images are written to the file.

/FACILITY=facility-name
/NOFACILITY (D)
Specifies the name of the facility to be monitored. This is only meaningful if at least one facility counter is displayed.

/IDENTIFICATION=process-id
/NOIDENTIFICATION (D)
Specifies the hexadecimal process-id of the process to be monitored. This is only meaningful if at least one process counter is to be displayed.

/INTERVAL[=delay-seconds]
/INTERVAL=2 (D)
Specifies how frequently RTR updates the screen. Delay-seconds is the number of seconds that RTR waits after completing one screen update before starting the next. Note that the interval between updates will always be slightly longer than Delay-seconds, depending on the complexity of the display and the number of nodes being monitored.

/LINK=link-name
/NOLINK (D)
Specifies the node name for the link to be monitored. This is only meaningful if at least one link counter is to be displayed.

/NODE=node-list
/NODE=default-node-list (D)
Specifies the names of the nodes to be monitored. If node-list is omitted then the local node is monitored.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/PARTITION=partition-name
/NOPARTITION (D)
Specifies the names of the partitions to be monitored. Partitions names have the form node$facility$partition-id.

/RESUME
/NORESUME (D)
Re-executes the last MONITOR command. The qualifiers /OUTPUT, /INTERVAL and /COUNT may be used with /RESUME. All other qualifiers are ignored. This

qualifier can be used to reset all the averages currently being displayed. It is also useful if monitoring is resumed after issuing one or more RTR commands.

/VERIFY
/NOVERIFY (D)
Specifies that the contents of monitor-file-spec are echoed on stdout. This is useful when developing monitor files to find the exact location of syntax errors.

Related Commands

- SCROLL
- PRINT
- CLEAR
- DISPLAY NUMERIC
- DISPLAY BAR
- DISPLAY TEXT

Examples

```
RTR> MONITOR CALLS/NODE=(TR2,TR1)/INTERVAL=10 1
RTR> SHOW PROCESS  2
RTR> MONITOR/RESUME  3
```

1 Display the CALLS picture, monitoring nodes TR2 and TR1 every ten seconds.

2 The SHOW PROCESS command is entered, interrupting the display.

3 Redisplay the CALLS picture using the original parameters.

```
RTR> MONITOR TRAFFIC/COUNT=10/OUTPUT=PICTURE.LIS
```

This command stores 10 images of the TRAFFIC picture in the file PICTURE.LIS.

QUIT

Quits from the RTR prompt.

Format

QUIT

Description

The **QUIT** command exits from the RTR prompt and returns control to the operating system prompt. The command has no parameters or qualifiers. Same as **EXIT**.

RECALL

Display a previously entered command for subsequent command editing.

Format

RECALL [command-specifier]

Command Qualifiers	Defaults
/ALL	/NOALL

Description

When you enter commands to the RTR Utility, they are stored in a recall buffer for later use with the RECALL command. Commands can be recalled by either entering the first few characters of the command or the command's number. The RECALL/ALL command can be used to list the last twenty commands.

When you recall a command, the RTR Utility displays the command but does not execute it. If you want to execute the command as it appears, press RETURN. You can also use the command editing facility to make changes in the command line and then press RETURN to process the revised version of the command.

Parameters

command-specifier
Specifies either the command number or the first few characters of the command you want to recall.

If command-specifier is omitted then the most recently entered command is recalled.

Qualifiers

/ALL
/NOALL (D)
Displays all the commands (and their numbers) available for recall.

Examples

```
RTR> CREATE FACILITY QUOTES/FRONT=FE3/ROUTER=TR2 1
RTR> SHOW FACILITY/LINK  2
RTR> RECALL CREATE  3
RTR> CREATE FACILITY QUOTES/FRONT=FE3/ROUTER=TR2  4
RTR> CREATE FACILITY ORDERS/FRONT=FE3/ROUTER=TR2 5
```

1 Create facility "QUOTES"

2 Check the links

3 Recall the CREATE FACILITY command

4 Change the facility name to "ORDERS" and resubmit the command.

REGISTER RESOURCE MANAGER (REGISTER RM)

Registers an instance of a resource manager (RM) with RTR.

Format

REGISTER RESOURCE MANAGER [resource_name]

REGISTER RM [resource_name]

Command Qualifiers	Defaults
/open_string	/see the Oracle8 Administrator's Reference manual
/close_string	/see the Oracle8 Administrator's Reference manual
/switch_name	/see the Oracle8 Administrator's Reference manual
/library_path	/path to an XA library
/protocol	/XA

Description

The REGISTER RESOURCE MANAGER command registers multiple resource managers or instances of resource managers (up to 16) with the current transaction manager. A different resource manager (RM) instance name is needed for each referenced database. Use this command after RTR ACP is started and before RTR facilities that reference this resource manager are created.

Note

This command is available only on UNIX and Windows NT systems.

Refer to Appendix C, XA Support for support information about XA.

Parameters

resource_name
Specifies the name of the resource to be registered.

Any application program which uses this resource must specify the same name when it calls `rtr_open_channel( )`.

Resource names can contain up to thirty characters. Letters, numbers and underline characters are all valid, but the first character of a resource name must be a letter.

The default value for `resource_name` is `RTR$DEFAULT_RESOURCE`.

Related Commands

- `UNREGISTER RESOURCE MANAGER`
- `SHOW RESOURCE MANAGER`

Examples

```
RTR> REGISTER RM rmi_1/open_string="Oracle_XA+Acc=P/user/pw+SesTm=15+db=accounting"
/close_string="" /xaswitch_name=xaosw /library_path="library_path"
```

SCROLL

Scroll a monitor picture.

Format

SCROLL direction [amount]

Description

The SCROLL command causes the the last picture that was displayed using the MONITOR command to be scrolled in the direction specified and then redisplayed.

Parameters

direction
Specifies the direction in which the screen is to be scrolled. Can be one of LEFT, RIGHT, UP, DOWN or HOME.

HOME scrolls the picture so that its top left corner coincides with the top left corner of the screen.

amount
Specifies the number of rows/columns by which the screen is scrolled. Amount is ignored if direction is specified as HOME.

Related Commands

- MONITOR

Examples

```
RTR> MONITOR TPS/INTERVAL=10) 1
RTR> SCROLL UP 10  2
RTR> SCROLL HOME  3
```

1 Display the TPS picture. This picture displays each process using RTR on a separate line. If there is insufficient space on the screen to display them all, the SCROLL command can be used to view a different portion of the list of processes.

2 Scroll the picture up 10 lines. Note that SCROLL automatically redisplays the current picture.

3 Restore the original picture position.

SET ENVIRONMENT

Specify the node(s) where subsequent RTR commands are executed.

Format

SET ENVIRONMENT

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE=node-list	/NODE=this_node

Description

The SET ENVIRONMENT command causes subsequent RTR commands to be executed on the specified nodes. Entering SET ENVIRONMENT without any qualifiers causes subsequent RTR commands to be executed on the local node only.

Qualifiers

/CLUSTER
Currently only fully supported in an OpenVMS environment, in which case it specifies that subsequent RTR commands are executed on all nodes in the VMScluster.

For other implementations, using /CLUSTER is interpreted as /NODE=this_node

/NODE[=node-list]
Specifies that subsequent RTR commands are executed on all nodes specified in node-list. If node-list is omitted or both the /NODE and /CLUSTER qualifiers are omitted then subsequent commands are executed on only the local node.

Related Commands

- SHOW ENVIRONMENT

Examples

See Section 1.5, Remote Commands, for examples of how to use the SET ENVIRONMENT command.

SET FACILITY

Sets various facility related options.

Format

SET FACILITY facility-name

Command Qualifiers	Defaults
/BROADCAST_MINIMUM_RATE=Bps	/BROADCAST_MINIMUM_RATE=1000
/QUORUM_THRESHOLD=n	/QUORUM_THRESHOLD=0
/BALANCE	/NOBALANCE
/CLUSTER	/NOCLUSTER
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/REPLY_CHECKSUM	/NOREPLY_CHECKSUM

Description

The SET FACILITY command sets the router load balancing and quorum characteristics of a facility.

Qualifiers

/BROADCAST_MINIMUM_RATE=Bps
/BROADCAST_MINIMUM_RATE=1000
/BROADCAST_MINIMUM_RATE=nnnn specifies the minimum rate (in bytes per second) to which flow control can reduce broadcast traffic on outgoing facility links from the node concerned.

For example, consider a facility has 100 frontends and 99 of them are able to receive data at a rate of 2KB per second, but one frontend has become congested and is not able to receive any. This can result in all the frontends slowing down to the rate that the slowest can accept.

Specifying /BROADCAST_MINIMUM_RATE=2000 on the router ensures that the 99 frontends receive their broadcasts, and that RTR attempts to send the broadcasts to the congested frontend. However, broadcasts for the congested frontend are discarded (if absolutely necessary) rather than slowing down all frontends to the rate that the slowest can accept.

Specifying /BROADCAST_MINIMUM_RATE without a value gives a default 1000 bytes per second; if you do not use the qualifier then the minimum is zero.

/QUORUM_THRESHOLD=n
/QUORUM_THRESHOLD=0
/QUORUM_THRESHOLD=n sets the minimum number of nodes/role combinations that have to be reachable to declare the configuration quorate.

Note

A node that combines both backend and router roles is counted twice in determining the threshold. A value of zero implies that the RTR determined threshold (half the number of node/role pairs configured plus one) is used. This is the default value; do not alter it unless you are sure that the unreachable nodes are really down. Before the rest of the nodes are started, it is recommended that this value is reset back to zero - the default setting.

The current value of quorum_threshold can be displayed with the SHOW FACILITY /STATE command.

/BALANCE
/NOBALANCE
/BALANCE specifies whether router load balancing is to be performed.

The default behavior (/NOBALANCE) is for a Frontend to connect to the preferred Router. Preferred Routers are selected in the order specified in the /ROUTER qualifier of the CREATE FACILITY command. This preference is subject to the Router being available and quorate. See Section 2.6, Router Load Balancing, for more information on load balancing.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/REPLY_CHECKSUM
/NOREPLY_CHECKSUM (D)
Specifies that the reply consistency check (or Response Matching) feature for replayed messages is enabled. It is a check for reply consistency during a replay of a reply to client message.

RTR can enable, disable and display this feature.

Related Commands

- SHOW FACILITY

Examples

```
 RTR> SET FACILITY FINANCE/QUORUM_THRESHOLD=4
quorum threshold set to 4 (from 0) for facility FINANCE
```

The SET FACILITY command tells RTR to set the quorum threshold to four for facility FINANCE. This command should be used on all the backend and router nodes in the facility.

```
 RTR> SET FACILITY FINANCE/BALANCE
```

This command tells RTR to use router load balancing.

SET LINK

Sets various link related options.

Format

SET LINK link-name

Command Qualifiers	Defaults
/AUTOISOLATE	/NOAUTOISOLATE
/ENABLE	/DISABLE
/CHECKSUM	/NOCHECKSUM
/CLUSTER	/NOCLUSTER
/INACTIVITY_TIMEOUT[=secs]	/INACTIVITY_TIMEOUT=node-default
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/SUSPECT	/NOSUSPECT

Description

The SET LINK command sets options for one or more or links. The options are link enabled or disabled, link autoisolate, link checksum and link inactivity timeout.

The parameter link-name is the node from which connect attempts are to be honored (or not). Note that disabling the link prevents incoming connections over an established link. It only takes effect when new connect attempts are made. It does not effect the ability to connect to a node whose link has been disabled. The link-name can be wildcarded.

The current state of the link can be displayed with the SHOW LINK/STATUS command. When looking at connection problems both ends of the link counters should be used with the SHOW LINK/COUNTER command.

Qualifiers

/AUTOISOLATE
/NOAUTOISOLATE (D)
Any RTR node may disconnect a remote node if it finds the remote node is unresponsive or congested. The normal behavior following such action is automatic network link reconnection and recovery.

Node autoisolation allows a node (the isolator) to disconnect a congested or unresponsive remote node (the isolatee) in such a way that when the congested node attempts to reconnect it receives an instruction to close all its network links and cease connection attempts. A node in this state is termed **isolated**.

Some applications require that a node which is suspected of causing congestion (that is, not processing network data sufficiently quickly) is isolated from the rest of the network, so as to cause minimum disruption. The node autoisolation feature meets this requirement.

Remote node autoisolation may be enabled (at the isolator) where it applies to all links using SET NODE/AUTOISOLATE, or for specific links only with the SET LINK/AUTOISOLATE command. An isolated node (isolatee) remains isolated until you carry out both of the following actions:

- Enable the link to the isolated node on all nodes that have isolated it, that is `set link [isolated-node]/enable`
- Exit the isolated state on the isolated node, that is `set node/noisolate`

Autoisolation is disabled (at the isolator) using the /NOAUTOISOLATE qualifier.

/CHECKSUM
/NOCHECKSUM (D)
`/CHECKSUM` specifies that checksum calculations for data packets over network links are performed. This qualifier is by default set to /NOCHECKSUM.

This command is useful for diagnosing errors over network links. To see the checksum state, use the SHOW LINK/STATE command.

/ENABLE
/DISABLE
`/ENABLE` specifies that connect attempts are honoured from the node specified by `link-name`.

This command is used to enable a link which is in a disabled state. A link can be disabled either as a result of operator action, or automatically if it has been suspected of causing severe congestion. If a link is automatically disabled, an entry is made in the RTR error log.

`/DISABLE` specifies that connect attempts are no longer honoured from the node `link-name`. Note that disabling the link does not have any immediate effect on an established link. It only takes effect when new connection attempts are made.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither `/NODE` nor `/CLUSTER` is specified then the command is executed on the nodes specified by the latest `SET ENVIRONMENT` command. If no `SET ENVIRONMENT` command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/INACTIVITY_TIMEOUT[=secs]
/INACTIVITY_TIMEOUT=node-default
`/INACTIVITY_TIMEOUT[=secs]` specifies the maximum elapsed time in seconds before RTR discards a link that is neither receiving traffic nor responding to explicit link state queries. Link failover occurs between the adjustable environmental timer parameters `RTR_TIMEOUT_CONNECT`, default of 60 seconds, and `RTR_TIMEOUT_CONNECT_RELAX`, default of 90 seconds, on a network link or remote node. When there is a failure RTR detects it within the timer parameters stipulated and disconnects and retries the link according to the router preferences for a frontend. If a router fails to respond to the reconnect tries there will be a time lapse of `RTR_TIMEOUT_CONNECT` plus `RTR_TIMEOUT_CONNECT_RELAX` for the link failover to occur.

The new value for secs becomes effective only after a time of about one third of the current value of the link inactivity timeout.

The minimum useful value for secs is three. If a value is not specified, links inherit the current value of the node inactivity timeout. (See SET NODE /INACTIVITY_TIMEOUT.)

You can check the current value of the link inactivity timeout with the command SHOW LINK linkname/COUNTER=ndb_lw_inact.

You should not specify a value of less than five times the time required for a round trip over the link. If you don't know this value, RTR can measure it for you. Make sure that there is no transactional traffic over the link, and monitor the link (with the MONITOR LINK command) between the two nodes whose round trip time you want to measure. After a few minutes, look at the link counters ndb_lw_trips and ndb_lw_trips_ms using the SHOW LINK /COUNTER=ndb_lw_trips* command. Dividing the latter by the former yields the average round trip time in milliseconds.

Note

The inactivity timeout is used for all RTR links, but the effect of a timeout and failover depends on what connections the link is supporting. In brief, a link between a router and a backend timing out causes a router or backend failover and quorum re-negotiations. A frontend will search for another router.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/SUSPECT
/NOSUSPECT
Obsolete. Available for compatibility reasons only, use /AUTOISOLATE instead.

Related Commands

- SET NODE/AUTOISOLATE
- SHOW LINK/STATUS

Examples

```
RTR> SET LINK JOEY/ENABLE
```

This command re-allows connections from node JOEY.

```
RTR> SET LINK JOEY/AUTOISOLATE
```

This command sets the autoisolate attribute on the link to node JOEY.

SET LOG

Specify where RTR writes its log messages.

Format

SET LOG

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/FILE=file-spec-list	/NOFILE
/NODE=node-list	/NODE=default-node-list
/OPERATOR	/NOOPERATOR
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SET LOG command specifies where RTR writes its log messages. You can write log messages to the operator console and to a maximum of four log files. Log files must be periodically purged to avoid difficulties with full disks. Do this by using SET LOG to specify a new file and deleting the old one.

If neither the /OPERATOR nor the /FILE qualifier is specified then logging is suppressed.

Note

Log files must always be accessible even if a node fails.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FILE=file-spec-list
/NOFILE (D)
Specifies the names of up to four files where the log messages are written. The given filename is appended with ".log".

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OPERATOR
/NOOPERATOR (D)
Specifies that messages are written to the operator log.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- SHOW LOG

Examples

```
RTR> SET LOG/FILE=RTRLOG.LOG/OPERATOR
```

This command tells RTR to write log messages to the file RTRLOG.LOG and to the operator log.

```
RTR> SET LOG/NOFILE/NOOPERATOR
```

This command suppresses all RTR log messages.

```
RTR> SET LOG/FILE="/usr/users/rtruser/daily_logfile"
```

This command tells RTR to write log messages to the file /usr/users/rtruser/daily_logfile.log.

```
RTR> SET LOG/FILE=("logfile1.log","logfile2.log")
```

This command tells RTR to write log messages to logfile1.log and logfile2.log.

```
RTR> SET LOG/OPERATOR
```

This command tells RTR to write log messages to the system log.

SET MODE

Specify whether RTR should run in a group mode or the nogroup (system) mode.

Format

SET MODE

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/GROUP[=user-id]	/NOGROUP
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SET MODE command specifies whether RTR runs in group mode or nogroup mode. (Nogroup mode is the same as system mode.)

Production systems use RTR in the default mode, i.e. nogroup (or system) mode, whereby all users running in this mode use one common invocation of RTR.

Developers typically wish to have their own invocation of RTR to avoid their RTR actions affecting other developers or the production system. This mode is termed group mode. Group mode allows development or testing of applications by several groups of people on the same system without interference.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/GROUP[=user-id
/NOGROUP
/GROUP specifies that RTR be set to GROUP mode for the user who issues the command. The groupname defaults to the first eight characters of your current user-id. You may also change to another group by entering a user or group ID. Note that group names are used for naming RTR journal files; do not use a group name containing the string "SYSTEM" or conflicts may occur.

```
RTR> set mode/group=develpr
```

/NOGROUP sets RTR into NOGROUP mode.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- SHOW MODE

Examples

```
RTR> SET MODE/GROUP
```

This command tells RTR to enter GROUP mode.

```
RTR> SET MODE/NOGROUP
```

This command tells RTR to enter NOGROUP mode.

SET NODE

Sets various node related options.

Format

SET node

Command Qualifiers	Defaults
/AUTOISOLATE	/NOAUTOISOLATE
/CLUSTER	/NOCLUSTER
/INACTIVITY_TIMEOUT[=secs]	/INACTIVITY_TIMEOUT=60
/ISOLATE	/NOISOLATE
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SET NODE command sets the automatic isolation characteristics and the link timeout default of a node.

Qualifiers

/AUTOISOLATE
/NOAUTOISOLATE (D)
Any RTR node may disconnect a remote node if it finds the remote node is unresponsive or congested. The normal behavior following such action is automatic network link reconnection and recovery.

Node autoisolation allows a node (the isolator) to disconnect a congested or unresponsive remote node (the isolatee) in such a way that when the congested node attempts to reconnect it receives an instruction to close all its network links and cease connection attempts. A node in this state is termed **isolated**.

Some applications require that a node which is suspected of causing congestion (that is, not processing network data sufficiently quickly) is isolated from the rest of the network, so as to cause minimum disruption. The node autoisolation feature meets this requirement.

Remote node autoisolation may be enabled (at the isolator) where it applies to all links using SET NODE/AUTOISOLATE, or for specific links only with the SET LINK/AUTOISOLATE command. An isolated node (isolatee) remains isolated until you carry out both of the following actions:

- Enable the link to the isolated node on all nodes that have isolated it, that is set link [isolated-node]/enable
- Exit the isolated state on the isolated node, that is set node/noisolate

Autoisolation is disabled (at the isolator) using the /NOAUTOISOLATE qualifier.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/INACTIVITY_TIMEOUT[=secs]
/INACTIVITY_TIMEOUT=60
/INACTIVITY_TIMEOUT[=secs] specifies the link inactivity timeout value for all current links, and also sets the default inactivity timeout value for new links. The default value is 60 seconds.

See SET LINK/INACTIVITY_TIMEOUT for further information.

/ISOLATE
/NOISOLATE (D)
The /ISOLATE qualifier is obsolete, and is available for compatibility reasons only. Use /AUTOISOLATE instead.

Use /NOISOLATE to take a node out of the isolated state. (See the /AUTOISOLATE qualifier for further information).

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- SHOW NODE
- SET LINK

Examples

```
RTR> SET NODE /NOISOLATE
```

This command tells RTR to set the executing node non-isolated.

SET PARTITION

Sets various partition related options.

Format

SET PARTITION partition-name

Command Qualifiers	Defaults
/FACILITY=[facility_name]	
/FAILOVER_POLICY=[SHADOW | STAND_BY]	
/IGNORE_RECOVERY	/NOIGNORE_RECOVERY
/PRIORITY_LIST=backend-node-list	/
/RECOVERY_RETRY_COUNT=n	/
/RESTART_RECOVERY	/
/RESUME	/
/SHADOW	/NOSHADOW
/SUSPEND	/
/TIMEOUT=nn	/

Description

The SET PARTITION command sets the characteristics of a named partition. Only backend partitions may be manipulated with this command; the command must be entered on the backend where the partition is located.

Use SET PARTITION any time after the partition has been created (either explicitly by CREATE PARTITION or implicitly by starting a server.) Note that the command only takes effect after the first server joins a partition. Any errors encountered at that time will appear as RTR log file entries. Using SET PARTITION to change the state of the system results in a log file entry.

Parameters

partition-name
The name of the partition being manipulated. This may be specified as partition_name (if the partition name is unique on the node), or as facility_name:partition_name.

Qualifiers

/FACILITY=facility_name **(D)**
Determines the facility that the partition command will act on. This is required.

/FAILOVER_POLICY=SHADOW
/FAILOVER_POLICY=STAND_BY (D)
Determines the action to take when the primary partition fails. The default action is to allow a standby of the primary to become the new primary. Optionally, RTR can be set to change state so that the secondary becomes primary, and a standby of the old primary (if any) becomes the new secondary.

/IGNORE_RECOVERY
/NOIGNORE_RECOVERY (D)
Forces the partition to exit any current wait state it may be in.

If a partition should enter a wait state or fail because of the unavailability of either a local or remote journal, this command can be used to override the default RTR behaviour. It instructs RTR to skip the current step in the recovery process. Since this command bypasses parts of the recovery cycle, use it with caution, and only when availability is more important than consistency in your application databases.

/PRIORITY_LIST=(backend-node-list)
Sets a relative priority that is used by RTR when selecting a backend member to make active. Enter a list of the backends in your configuration in decreasing order of priority; the relative order of the list will be taken into consideration when RTR is determining on which node to make a partition active.

It is not an error to enter different versions of the priority list at different backends, but this is not recommended. It is recommended to suspend partitions before changing the priority list.

/RECOVERY_RETRY_COUNT=n
The recovery retry count indicates the maximum number of times that a transaction should be presented for recovery before being written to the journal as an exception. Once a transaction has been recorded as an exception, it is no longer considered eligible for recovery and will require manual processing by a qualified individual.

The recovery retry count is partition specific, and applies to both local and shadow recovery operations. The default is no limit on the number of retries, which permits a killer message to bring down all available servers servicing a given partition.

The recovery retry count should be set prior to starting (or restarting) the application servers so that the limit is established prior to the start of recovery operations.

/RESTART_RECOVERY
Restarts the recovery cycle. The recovery cycle can be manually restarted with /RESTART. This is useful if the operator previously aborted recovery through use of /IGNORE_RECOVERY. Since this command may result in recovery of transactions from previously inaccessible journals, it should not be used if your applications are sensitive to the order in which transactions are processed by the servers.

/RESUME
Resume normal operations (that is, cancel a /SUSPEND command). If the partition is already in the desired state, the command has no effect.

/SHADOW
/NOSHADOW
Turns shadowing on or off for the specified partition.

Shadowing for a partition can be turned off only in the absence of an active secondary site, that is, the active member must be running in remember mode. The command will fail if it is entered on either an active primary or secondary. If entered on a standby of either the primary or secondary, the command is accepted but fails in the RTR router; this is recorded with a log file entry at the router.

Once shadowing is disabled, the secondary site servers will be unable to startup in shadow mode until shadowing is enabled again.

Shadowing for the partition can be turned on by entering the command at the current active member or any of its standbys.

If shadowing is already in the desired mode the command has no effect.

/SUSPEND
Stops presenting new transactions on the specified partition until a /RESUME is issued.

Use /SUSPEND to temporarily halt the presentation of new transactions to servers on the backend where the command is entered. The command completes when the processing of all currently active transactions is complete.

The optional /TIMEOUT qualifier may be used to limit the time that the command waits for completion. If the command times out, presentation of new transactions is suspended, but there still exist transactions for which servers have yet to complete processing. The operator must decide to either reenter the command and wait a further period of time, or resume the partition. Note that use of this command does not affect any transaction timeout value specified by RTR clients, so such transactions may encounter a timeout condition if the partition remains suspended.

/TIMEOUT
See description for /SUSPEND.

Related Commands

- CREATE PARTITION
- SHOW PARTITION

SET TRANSACTION

Sets various transaction related options.

Format

SET TRANSACTION transaction-id

Command Qualifiers	Defaults
/BEFORE[=date]	today
/STATE=current_state	
/FACILITY=facility_name	/RTR$DEFAULT_FACILITY (D)
/NEW_STATE=new_state	
/NODE[=node_list	/default_node_list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/PARTITION=partition_name	
/SINCE[=date]	today
/USER=username	all users

Description

Note

The SET TRANSACTION command could damage your journal and database integrity if used incorrectly. Ensure that you fully understand the reason and impact for changing a transaction state before altering your RTR system.

The SET TRANSACTION command allows you to modify the state of a transaction or set of transactions stored in the RTR journal.

This command is complementary to the DUMP JOURNAL and SHOW TRANSACTION commands in that it gives capability of reading and modifying the status of a transaction status in the RTR journal. For example, if a shadow recovery is known to be unnecessary, you may want to clean up the RTR journal to prevent the committed transactions in the journal from being replayed. Using the SET TRANSACTION command, the RTR administrator is able to delete that set of transactions from the journal.

In addition, the SET TRANSACTION command also helps users to better control the RTR runtime environment in difficult operational situations. For example, a transaction while still in SENDING state on the backend may appear to be hung and cannot proceed. Using the SET TRANSACTION command, an RTR administrator can abort this transaction "on-the-fly" and free runtime resources.

While the SET TRANSACTION command enables RTR users to have full control of RTR transactions, it introduces the risk that a transaction could be lost or corrupted. The command must be used with discretion and should only be used by expeerienced RTR system administrators.

After using the SET TRANSACTION command complete, you may use the DUMP JOURNAL command to verify the results.

Usage Notes

The command can only be executed on a backend node in which the journal is located and the RTR log file must be turned on to record the transaction changes. RTR needs to be started before using this command.

When a transaction's state is changed, the new state is written to the RTR journal synchronously. RTR will try to determine whether the change also affects other portion of RTR environment. For example, in a runtime situation where RTR routers and RTR backend servers are active, the RTRACP will send the new status to applicaiton servers as well as RTR routers to make sure that the change takes effect on all nodes in the RTR facility or facilities.

However, in an off-line situation where an RTR facility has not been created, RTR will simply update the transaction state in place in the RTR journal. The RTR log file must be turned on before using the SET TRANSACTION command to record the state changes. See the SET LOG command.

There are eight legitimate situations where you can change a transaction's state; See Table 6–19.

Parameters

transaction-id

Specifies a particular transaction or transactions whose transaction state you want to change. The transaction_id format is the same as that displayed by DUMP JOURNAL and SHOW TRANSACTION commands.

If no transaction_id is specified then all transactions ("*") that satisfy the specifying qualifiers are processed by the command.

Qualifiers

/BEFORE[=date]

Selects only those transactions whose timestamp is before the specified date. Default is the current date.

/STATE=current_state

Selects a particular transaction or a set of transactions that are in the specified state transaction state. This qualifier is required and the state value must be specified.

Value of state may be one of the following:

SENDING
VOTED
COMMIT
EXCEPTION
PRI_DONE

Use the DUMP JOURNAL or SHOW TRANSACTION command to help you find what is the current state of a particular transaction.

/FACILITY
/FACILITY=RTR$DEFAULT_FACILITY (D)

Specifies the name of a facility for selecting transactions. The default facility, RTR$DEFAULT_FACILITY, is used if this qualifier is not specified.

/NEW_STATE
Specifies the new transaction state that selected transactions will be changed to. This qualifier is required and new state value must be specified.

Value of new_state may be one of the following:

ABORT
COMMIT
DONE
EXCEPTION

Note that one cannot always change a transaction's state from one legitimate transaction state to another. Some state changes are not valid. The following table shows state changes that are valid.

Table 6–19 Valid Transaction State Changes

	NEW STATE			
Current State	**COMMIT**	**ABORT**	**EXCEPTION**	**DONE**
SENDING		YES		
VOTED	YES	YES		
COMMIT			YES	YES
EXCEPTION	YES			YES
PRI_DONE				YES

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/PARTITION
/PARTITION=partition_name
Specifies the name of a partition from which all transactions are running within. A partition name must be supplied.

Use SHOW PARTITION to view the names of the currently active partitions.

/SINCE[=date]
Selects only those transactions whose timestamp is after the specified date. Default is the current date.

Related Commands

- SHOW TRANSACTION
- DUMP JOURNAL

Examples

```
RTR> SET TRANSACTION "50d01f10,0,0,0,0,2166,522b2001" -
_RTR> /NEW=ABORT /CURRENT=SENDING /PART=DB_PART
```

Abort this specified transaction running in the DB_PART partition.

```
RTR> SET TRANSACTION /NEW=ABORT /CURRENT=VOTED /PART=DB_PART
```

For all transactions that are in VOTED transaction state and are running in DB_PART partition and in "RTR$DEFAULT_FACILITY facility, abort them.

SHOW CHANNEL

Show the names and state of channels that have been opened using the CLI API.

Format

SHOW CHANNEL [channel-name]

Command Qualifiers	Defaults
/ALL_WINDOWS	/NOALL_WINDOWS
/CLUSTER	/NOCLUSTER
/NODE=node-list	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SHOW CHANNEL command shows the channel type (client or server), the channel name and owner process-id for channels opened using the CLI API.

Parameters

channel-name
Specifies the name of the channel to be displayed. channel-name can contain wild cards. If channel-name is omitted all declared channels for this window (terminal or virtual terminal) are displayed.

Qualifiers

/ALL_WINDOWS
/NOALL_WINDOWS
Specifies that channels opened in all windows (terminals or virtual terminals) are shown.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- call rtr_open_channel( )
- call rtr_close_channel( )

Examples

```
RTR> SHOW CHANNEL/ALL_WINDOWS  1

Channel type Channel name               (Owner pid)
server         RTR$DEFAULT_CHANNEL            (28879)  2
client         CLI_CHN                        (28879)  3
client         CLI_CHN2                       (26225)  4
```

1 Display information about all declared channels.

2 The channel called RTR$DEFAULT_CHANNEL is open as a server channel.

3 The channel called CLI_CHN is open as a client channel.

4 The channel called CLI_CHN2 is open as a client channel, and has been opened by another process (in another window).

SHOW CLIENT

Display information about client channels.

Format

SHOW CLIENT

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/FACILITY	/FACILITY="*"
/FULL	none
/IDENTIFICATION=process-id	/NOIDENTIFICATION
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SHOW CLIENT command displays information about client channels.

Information such as PID, key range, state, event mask and event name are displayed.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/IDENTIFICATION=process-id
/NOIDENTIFICATION (D)
Specifies the PID of the process for which information is displayed. The default (/NOIDENTIFICATION) displays information for all clients.

/FACILITY
/FACILITY="*" (D)
Specifies the facility name for which information should be displayed.

By default, information is displayed for all Facilities.

/FULL
none (D)
Specifies a detailed listing of client information.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Examples

```
RTR> SHOW CLIENT/FULL  1

Process-id:      9234          Facility:       TEST43  2
Channel:         500           Flags:             CLI  3
State:           declared      rcpnam:       "V3TEST" 4
User Events:     255           RTR Events:          0 5
```

1 Show the client in detail.

2 The client's process id and facility name.

3 The client's channel id and the flags set when the channel was opened.

4 The channel state and recipient name for events.

5 The user events and RTR events subscribed to.

SHOW DISPLAY

Show which items were displayed by the most recently issued MONITOR command or DISPLAY commands.

Format

SHOW DISPLAY

Command Qualifiers	Defaults
/ALL	/NOALL
/OUTPUT[=file-spec]	/OUTPUT=stdout
/X=column	
/Y=row	

Description

The SHOW DISPLAY command shows which items were displayed by the most recently issued MONITOR command. These may have been read in from a display file using the MONITOR file-spec command, or entered interactively using DISPLAY commands.

The item definitions are shown in DISPLAY command format. (The command SHOW DISPLAY/OUTPUT=file may be used to create new monitor files.)

Qualifiers

/ALL
/NOALL (D)
Specifies that all monitored items are shown.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/X=column /Y=row
Specify that the single item in position (column,row) is shown.

Related Commands

- MONITOR
- DISPLAY NUMERIC
- DISPLAY BAR
- DISPLAY TEXT

Examples

```
RTR> MON CALLS  1
RTR> SHOW DISPLAY/ALL  2

DISPLAY TEXT  "RTR api calls, Node: $node_name ,- 3
         PID: $process_id, Process name: -ALL-" -
        /X=1   /Y=1 -
        /BOLD="1"
   .
   .
   .
DISPLAY NUMERIC         "rtr_open_channel_succ" -
        /X=1   /Y=5 -
        /BLANK -
        /LABEL=" rtr_open_channel    " -
        /WIDTH=9
   .
   .
   .
```

1 Display the CALLS monitor picture.

2 Show all the items contained in the monitor picture.

3 The items are displayed in DISPLAY command format. See the description of the relevant DISPLAY commands for a description of the various qualifiers.

Examples

See Section A.1, Interactive Definition of a Monitor Picture, for an example of how to use the SHOW DISPLAY command.

SHOW ENVIRONMENT

Shows the default nodes used for remote command execution.

Format

SHOW ENVIRONMENT

Description

The SHOW ENVIRONMENT command shows which nodes are used by default for remote command execution.

Related Commands

- SET ENVIRONMENT

Examples

```
RTR> SET ENVIRONMENT/NODE=(FE2,FE3)  1
RTR> SHOW ENVIRONMENT  2
%RTR-S-COMARESEN, commands sent by default to node FE2 3
%RTR-S-COMARESEN, commands sent by default to node FE3
```

1 Set the command environment so that subsequent commands will be executed on nodes FE2 and FE3.

2 Show which nodes are selected.

3 The RTR-S-COMARESEN message is displayed for each selected node.

SHOW FACILITY

Show the names, configuration and status of one or more facilities.

Format

SHOW FACILITY [facility-name]

Command Qualifiers	Defaults
/BALANCE	/NOBALANCE
/CLUSTER	/NOCLUSTER
/CONFIGURATION	/NOCONFIGURATION
/COUNTER[=counter-name]	/NOCOUNTER
/FULL	/NOFULL
/LINKS	/NOLINKS
/STATE	/NOSTATE
/OUTPUT[=file-spec]	/OUTPUT=stdout
/NODE[=node-list]	/NODE=default-node-list

Description

The SHOW FACILITY command shows the names, configuration and status of facilities on the node where the command is executed. If no qualifiers are used, only the facility name or names and the roles in the facility on the node are displayed.

Parameters

facility-name
Specifies the name of the facility you want to display. Facility-name may contain wild cards ("*" and "%"), in which case all matching facilities will be displayed. If facility-name is omitted all configured facilities are displayed.

Qualifiers

/BALANCE
/NOBALANCE (D)
Specifies that the load-balancing status of the facility is also displayed. See Section 2.6, Router Load Balancing for details of load-balancing.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/CONFIGURATION
/NOCONFIGURATION (D)
Specifies that the facility configuration is to be displayed. The configuration information indicates the role(s) of the the node where the command is executed, and

Whether router call-out servers or backend call-out servers have been configured,
Whether load balancing has been configured,
Whether quorum check is being handled on this node.
Whether reply consistency check for replayed messages is enabled.

/COUNTER[=counter-name]
/NOCOUNTER (D)
Specifies that the facility counters are displayed. `Counter-name` is the name of the counter. If `counter-name` is omitted then all counters are displayed. `Counter-name` may contain wild card characters.

/FULL
/NOFULL (D)
Equivalent to specifying `/CONFIGURATION`, `/STATE` and `/LINK`.

/LINKS
/NOLINKS (D)
Lists all links to connected nodes in the facility (including the node where the command is executed. The following information is shown for each link:

Whether the node on this link is active as frontend, router or backend roles.
What kind link is active, that is,

frontend to router
router to frontend
router to backend
backend to router

Whether the router and backend roles are quorate,
Whether the router role is the current router for the frontend,
Whether the backend role is acting as a load balancing coordinator.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in `node-list`. If `node-list` is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file `file-spec`. If `/OUTPUT` or `file-spec` is omitted then the standard or default output is used.

/STATE
/NOSTATE(D)
Specifies that the state of the facility is displayed.

On nodes having a backend or router role, the quorum status is shown. Also, the current quorum threshold is shown (default quorum value is shown as zero), and the minimum broadcast rate (if it has been set).

On a frontend, this qualifier can be used to find out whether the node is currently connected to a router.

Related Commands

- CREATE FACILITY
- DELETE FACILITY
- SHOW LINK

Examples

```
 RTR> SHOW FACILITY/FULL/NODE=BRONZE  1

Facility:       FUNDS_TRANSFER  2
Configuration:-                 3

Frontend:         no    Router:              yes  Backend:            yes
                        Router call-out:      no  Backend call-out:    no
                        Load balance:         no  Quorum-check off:    no

State:-                         4

FE -> TR:      unused   Router quorate:      yes  Backend quorate:    yes
                        Quorum threshold:     0  Min broadcast rate:   0

Links:-                         5

Link to:                      bronze
Frontend:           no  Router:              yes  Backend:            yes
Router -> Frontend:no   Frontend -> Router:   no
                        Backend -> Router:   yes  Router -> Backend:  yes
                        Router quorate:      yes  Backend quorate:    yes
                        Router current:       no  Backend coordinator:yes

Link to:                      airola    6
Frontend:           yes Router:               no  Backend:             no
Router -> Frontend:yes  Frontend -> Router:   no
                        Backend -> Router:    no  Router -> Backend:   no
                        Router quorate:       no  Backend quorate:     no
                        Router current:       no  Backend coordinator: no
```

1 Show all facilities in detail.

2 The facility's name.

3 The facility's configured roles on this node. In this example, the node is a router and a backend; load balancing, router callouts and backend callouts are not enabled.

4 The facility's state. The router and backend roles are quorate. (Quorate means that this node's view of the availability of the other nodes in the configuration matches that of the other backends and routers. A facility that is not in the quorate state is effectively unusable.) The quorum threshold is at the default (zero) and no minimum broadcast rate has been set.

5 The facility's links. The first link shown (to node "bronze" i.e. itself) shows that it is a router, connected to the backend, and the router is quorate. The router is not current because there is no frontend on this connection. The link is also a backend, connected to the router, and this backend is quorate. This backend would also be the backend coordinator if load balancing was enabled.

6 The second link shown (to node "airola") shows that it is a frontend, and is a frontend to router connection.

```
RTR> SHOW FACILITY/BALANCE
Facilities:

Facility:       RTR$DEFAULT_FACILITY

Load:-

Frontends connected:              1     Frontends allowed:               1
Load coordinator:               yes     Quorate routers:                 1
Total Frontends:                  1     Current Credit:                  1
FE -> TR:                         -
```

This display shows the load balancing status. This node is the current load coordinator.

SHOW JOURNAL

Display information about current RTR journal files.

Format

SHOW JOURNAL

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/FILENAMES	/NOFILENAMES
/FULL	/NOFULL
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SHOW JOURNAL command shows the disks where the RTR journal files reside, and (optionally) the maximum and allocated number of blocks for each journal file and the journal file name.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FILENAMES
/NOFILENAMES (D)
/FILE adds the journal filenames to the display.

/FULL
/NOFULL (D)
/FULL adds to the display the maximum and allocated number of blocks for each journal file.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- CREATE JOURNAL
- DELETE JOURNAL
- INITIALIZE JOURNAL

Examples

```
RTR> SHOW JOURNAL/FULL/FILENAMES  1

RTR journal:-
    2          3             4         5           6
Disk(1):  /dev/rz3a      Blocks: 1500 Allocated: 1502 Maximum: 3000
File(1):  /dev/rz3a /rtrjnl/SYSTEM/BRONZE.J017
Disk(2):  /dev/rz2c      Blocks: 1500 Allocated: 1502 Maximum: 3000
File(2):  /dev/rz2c /usr/users/rtrjnl/SYSTEM/BRONZE.J11
```

1 Show the disks used for RTR's recovery journal and the filenames.

2 Two disks are currently in use.

3 Device name.

4 File size in blocks.

5 Allocated file size in blocks.

6 Maximum file size in blocks.

7 Filename.

SHOW KEY

Display the key definitions created by the DEFINE /KEY command.

Format

SHOW KEY [key-name]

Command Qualifiers	Defaults
/ALL	/NOALL
/FULL	/NOFULL
/OUTPUT[=file-spec]	/OUTPUT=stdout
/IF_STATE	/NOIF_STATE

Description

The SHOW KEY command shows the key definitions created by the DEFINE /KEY command.

Parameters

key-name
Specifies the name of the key whose definition you want displayed. See the description of the DEFINE /KEY command for a list of the valid key names.

Qualifiers

/ALL
/NOALL (D)
Requests that all key definitions in the current state be displayed. You can use the /IF_STATE qualifier to request key definitions in other states. Do not specify a key name with the /ALL qualifier. If no state is specified, all key definitions in the current state are displayed.

/FULL
/NOFULL (D)
Requests that all qualifiers associated with a definition are displayed. By default, only the state of the definition and the definition itself are displayed.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/IF_STATE=state-name
/NOIF_STATE (D)
Specifies the name of a state for which the specified key definitions are displayed. State names can be any appropriate alphanumeric string. State names are created with the DEFINE /KEY command. If you omit the /IF_STATE qualifier or use /NOIF_STATE, key definitions in the current state are displayed.

Related Commands

- DEFINE /KEY

Examples

```
RTR> SHOW KEY/FULL
DEFAULT PF1 defined as ""
DEFAULT KP0 defined as "MONITOR/RESUME"
DEFAULT KP2 defined as "SCROLL DOWN 1"
DEFAULT KP4 defined as "SCROLL LEFT 1"
DEFAULT KP5 defined as "SCROLL HOME"
DEFAULT KP6 defined as "SCROLL RIGHT 1"
DEFAULT KP8 defined as "SCROLL UP 1"
GOLD KP2 defined as "SCROLL DOWN 10"
GOLD KP4 defined as "SCROLL LEFT 10"
GOLD KP6 defined as "SCROLL RIGHT 10"
GOLD KP8 defined as "SCROLL UP 10"
```

SHOW LINK

Display the configuration and status of the links to other nodes.

Format

SHOW LINK [node-name]

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/COUNTER[=counter-name]	/NOCOUNTER
/FACILITY	/NOFACILITY
/FULL	/NOFULL
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/STATE	/NOSTATE

Description

The SHOW LINK command shows the configuration and status of the links to RTR nodes. If no qualifiers are given, a brief display of the links is shown.

Parameters

node-name
Specifies the name of a node; the parameters for the link to this node are displayed. Node-name may contain wild cards ("*" and "%"), in which case all matching links will be displayed. If node-name is omitted all known links are displayed.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/COUNTER[=counter-name]
/NOCOUNTER (D)
Specifies that the link counters are also to be displayed. Counter-name is the name of the counter to be displayed. If counter-name is omitted then all counters will be displayed. Counter-name may contain wild card characters.

/FACILITY
/NOFACILITY (D)
Specifies that the names of facilities using the link are also displayed.

/FULL
Equivalent to specifying /FACILITY/STATE.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/STATE
/NOSTATE (D)
Specifies that a detailed status of the specified link(s) is displayed.

Related Commands

- SET LINK
- CREATE FACILITY
- SHOW FACILITY/LINK

Example

```
RTR> SHOW LINK/FULL IRON
Links:

To Node:                        iron     Address:           iron.zuo.dec.com

Status:-

Outgoing message sequence nr:     22     Incoming message sequence nr:     22
Current receive buffer size:    2048     Current transmit buffer size:   2048
Current number of link users:      1     Write buffer timed out:           no
Write buffer full, may be sent:   no     Write buffer allocated:          yes
I/O error detected in write:      no     I/O error detected in read:       no
Pipe temporarily blocked:         no     Connection broken:                no
Write issued, not completed:      no     Read is pending:                 yes
Node initiated the connection:   yes     Connection established:          yes
Connection in progress:           no     Node is configured:              yes
Link is in disabled state:        no     Link may be suspect:              no

Facilities:-

In facility:                   steve2
Frontend:            no   Router:              yes   Backend:            no
Router -> Frontend:  no   Frontend -> Router:  yes
                          Backend -> Router:    no   Router ->Backend:   no
                          Router quorate:       no   Backend quorate:    no
                          Router current:      yes   Backend coordinator:no
```

This example shows

- The name and network address of the partner node.
- Link status.
- Name of facility using this link and how the link is used in the facility.

SHOW LOG

Display the names of the current log files.

Format

SHOW LOG

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SHOW LOG command shows the names of the current RTR log files as defined with the SET LOG command.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- SET LOG

Example

```
RTR> SHOW LOG  1
Messages not being sent to operator console  2
Log file[1]:    /usr/users/someone/rtr_logfile.log   3
```

1 Show where RTR log messages are currently written.

2 Currently not being sent to the operator log.

3 Currently being written to file `rtr_logfile.log`.

SHOW MODE

Displays the current RTR mode.

Format

SHOW MODE

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SHOW MODE command shows the currently running user group for RTR. For nogroup (system) mode, a null group name is displayed. SET MODE command for further information about modes.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- SET MODE

Examples

```
RTR> SHOW MODE

Group name is "develop"
```

SHOW NODE

Shows the node network status, the autoisolation state and the node inactivity timer.

Format

SHOW NODE

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SHOW NODE shows the network status, the autoisolation state (enabled or disabled) and the node inactivity timer.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- SET NODE/ISOLATE
- SET LINK

Examples

```
 RTR> SHOW NODE  1
Node properties:

Network state:              enabled 2
Auto isolation:            disabled  3
Inactivity timer/s:              60  4
```

1 SHOW NODE command.

2 Network status

3 Auto isolation state (enabled or disabled).

4 Inactivity timer value in seconds.

SHOW PARTITION

Display server data partition information.

Format

SHOW PARTITION

Command Qualifiers	Defaults
/BACKEND	/BACKEND
/BRIEF	/NOBRIEF
/CLUSTER	/NOCLUSTER
/FACILITY	/FACILITY="*"
/FULL	/NOFULL
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/ROUTER	/ROUTER

Description

The SHOW PARTITION command displays information about key range partitions, their states, and current transaction activity. This information is useful for diagnosing bottlenecks or partition difficulties in RTR.

The "States" field of SHOW PARTITION /BACKEND can display the following values:

Table 6–20 Key-Range States

State	Meaning
wt_tr_ok	Server is waiting for routers to accept it
wt_quorum	Server is waiting for backend to be quorate
lcl_rec	Local recovery
lcl_rec_fail	Primary server waiting for access to a restart journal
lcl_rec_icpl	Getting next journal to recover from
lcl_rec_cpl	Processed all journals for local recovery
shd_rec	Shadow recovery
shd_rec_fail	Shadow server waiting for access to a restart journal
shd_rec_icpl	Shadow getting next journal to recover from
shd_rec_cpl	Processed all journals for shadow recovery
catchup	Secondary is catching up with primary
standby	Server is declared as standby
active	Server is active
pri_act	Server is active as primary shadow
sec_act	Server is active as secondary shadow
remember	Primary is running without shadow secondary

The "State" field of SHOW PARTITION /ROUTER can display the following values:

Table 6–21 Router Partition States

State	Meaning
BLOCKED	Key range is recovering or awaiting journal access
ACTIVE	Primary server is ready to accept transactions
CATCHUP	Secondary server is catching up with primary
TAKEOVR	Standby take-over is in progress
LAGGING	Secondary is ready, primary is still recovering

Qualifiers

/BACKEND
/NOBACKEND (D)
Displays information about backend partitions; it shows the partition state and low and high bounds. /BACKEND with /FULL provides more detailed information for a partition, giving the current queue depth (transactions active) on a partition. This is useful for determining whether a server is processing transactions correctly. `Last Rcvy BE:` shows from which node recovery information may be fetched; `Txns Rcvrd:` shows the number of transactions actually recovered.

The default is to output brief router and backend information.

/BRIEF
/NOBRIEF (D)
Brief output gives a one line listing of each partition in the system, showing the partition name, the facility in which it resides, and the partition state. Since space on the line is limited, partition names my be truncated to fit.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither `/NODE` nor `/CLUSTER` is specified then the command is executed on the nodes specified by the latest `SET ENVIRONMENT` command. If no `SET ENVIRONMENT` command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FACILITY
/FACILITY="*" (D)
Specifies the facility name for which information should be displayed.

By default, information is displayed for all Facilities.

/FULL
/NOFULL (D)
Gives a detailed listing of server partition information.

The following items are displayed:

- Key segment low and high bounds
- Counts of the active and free servers bound to the partition
- Counts of the active and recovered transactions for the partition

- The state of transaction presentation - one of active, suspended or suspending
- The current failover policy - one of fail_to_standby, fail_to_shadow or pre_ v32_compatibilty

The SHOW PARTITION comand displays callout server data as backend server data because a callout server uses server, not router, data structures. A callout server actually runs on the router identified for its facility.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in `node-list`. If `node-list` is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file `file-spec`. If `/OUTPUT` or `file-spec` is omitted then the standard or default output is used.

/ROUTER
/NOROUTER (D)
Displays information about router partitions; it shows the partition state and partition low and high bounds. It can be used to quickly determine the current primary node for a given partition. Using /ROUTER with /FULL provides more detailed information, such as the main, shadow and standby servers for the partitions, as seen from the router.

Partition bounds are shown as ASCII characters if there is a translation, otherwise as hexadecimal digits. Only the least significant four bytes are shown.

Examples

The following example shows a router partition.

```
RTR> SHOW PARTITION/ROUTER

Facility                      Low Bound        High Bound       Primary Main
RTR$DEFAULT_FACILITY          0                4294967295       depth

RTR> SHOW PARTITION/ROUTER/FULL

Facility:       RTR$DEFAULT_FACILITY     State:                       ACTIVE
Low Bound:                         0     High Bound:               4294967295
Failover policy:                                                fail_to_standby
Backends:                                                                depth
States:                                                                  active
Primary Main:                  depth     Shadow Main:
```

The following examples show two backend partitions.

```
RTR> SHOW PARTITION/BACKEND

Partition name                    Facility                      State
RTR$DEFAULT_PARTITION_16777217    RTR$DEFAULT_FACILITY          active
RTR$DEFAULT_PARTITION_16777218    RTR$DEFAULT_FACILITY          active

RTR> SHOW PARTITION/BACKEND/FULL
```

```
Partition name:                          RTR$DEFAULT_PARTITION_16777217
Facility:       RTR$DEFAULT_FACILITY     State:                      active
Low Bound:                   "aaaa"      High Bound:                 "mmmm"
Active Servers:                 0        Free Servers:                  1
Transaction presentation:     active     Last Rcvy BE:
Txns Active:                    0        Txns Rcvrd:                    0
Failover policy:     fail_to_standby     Key range ID:           16777217

Partition name:                          RTR$DEFAULT_PARTITION_16777218
Facility:       RTR$DEFAULT_FACILITY     State:                      active
Low Bound:                   "nnnn"      High Bound:                 "zzzz"
Active Servers:                 0        Free Servers:                  1
Transaction presentation:     active     Last Rcvy BE:
Txns Active:                    0        Txns Rcvrd:                    0
Failover policy:     fail_to_standby     Key range ID:           16777218
```

SHOW PROCESS

Display information about processes which are using RTR.

Format

SHOW PROCESS

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/COUNTER[=counter-name]	/NOCOUNTER
/FULL	/NOFULL
/IDENTIFICATION=process-id	/NOIDENTIFICATION
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SHOW PROCESS command displays information about the processes using RTR.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/COUNTER[=counter-name]
/NOCOUNTER (D)
Specifies that the process counters are also to be displayed. counter-name is the name of the counter to be displayed. If counter-name is omitted then all counters will be displayed. counter-name may contain wild card characters.

/IDENTIFICATION=process-id
/NOIDENTIFICATION (D)
Specifies the process about which information is required. The default (/NOIDENTIFICATION) displays all processes using RTR.

/FULL
/NOFULL (D)
Equivalent to specifying /COUNTER.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Examples

```
RTR> SHOW PROCESS
Processes:

Process-id   Process Name
326
```

SHOW REQUESTER

See SHOW CLIENT. The SHOW REQUESTER command has been replaced by SHOW CLIENT and is retained for compatibility reasons only.

SHOW RESOURCE MANAGER (SHOW RM)

Displays resource manager instance (RM) information.

This command is available only on UNIX and Windows NT systems.

Format

SHOW RESOURCE MANAGER [resource_name]

SHOW RM [resource_name]

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/FULL	/NOFULL
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SHOW RESOURCE MANAGER command displays information for registered RMs. This command shows information for one or more RMs registered with the current TM.

Note

This command is available only on UNIX and Windows NT systems.

Refer to Appendix C, XA Support for support information about XA.

Parameters

resource_name
Specifies the name of the resource manager instance you want to display. resource_name may contain wild cards ("*" and "%"), in which case all matching resources will be displayed. If resource_name is omitted all configured resources are displayed.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FULL
/NOFULL (D)
Displays additional information for facilities that reference a particular RM. If no rmi_name is included, displays information for all RMs.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- REGISTER RESOURCE MANAGER
- UNREGISTER RESOURCE MANAGER

Examples

```
RTR> SHOW RM rmi_1 /full
     Resource Manager:
     RMI Name                : Nashua
     RMID                    : yyy
     XA Switch               : xaosw
     Library path            : zzz
     Facilities associated with this RM: Nashua
```

SHOW RTR

Display the configuration and status of RTR.

Format

SHOW RTR

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/COUNTER[=counter-name]	/NOCOUNTER
/FULL	/NOFULL
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/STATUS	/NOSTATUS
/VERSION	/NOVERSION

Description

The SHOW RTR command displays the configuration and status of RTR.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/COUNTER[=counter-name]
/NOCOUNTER (D)
Specifies that the per node counters are also to be displayed. Counter-name is the name of the counter to be displayed. If counter-name is omitted then all counters will be displayed. Counter-name may contain wild card characters.

/FULL
/NOFULL (D)
Equivalent to specifying /COUNTER/STATUS.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/STATUS (D)
/NOSTATUS
Displays the current status of RTR (started, stopped and so on).

/VERSION
/NOVERSION (D)
Displays the RTR version.

Related Commands

- STOP RTR
- START RTR

Example

```
RTR> SHOW RTR  1

RTR running on node baby.home.dec.com in group: develpr  2
```

1 Show the state and configuration of RTR.

2 RTR has been started in group mode for group "develpr".

SHOW SEGMENT

Display the type and size of routing key segments.

Format

SHOW SEGMENT

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/FACILITY	/FACILITY="*"
/FULL	/NOFULL
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SHOW SEGMENT command displays the routing key segment definitions.

The routing key definition is common to all servers in a facility.

The routing key specifies the data within a message sent from clients used to route the message to a particular key range server.

The position, length and data type of each key segment is displayed.

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FACILITY
/FACILITY="*" (D)
Specifies the facility name for which information should be displayed.

By default, information is displayed for all Facilities.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Examples

```
RTR> SHOW SEGMENT  1

Facility               Data Type          Length      Offset
RTR$DEFAULT_FACILITY   UNSIGNED           1           0       2
TEST_FAC               SIGNED             4           10
```

1 Show the routing key segments for all facilities.

2 The facility name, the routing key data type, key length and offset are shown for each facility.

SHOW SERVER

Display information about server channels.

Format

SHOW SERVER

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/FACILITY	/FACILITY="*"
/FULL	/NOFULL
/IDENTIFICATION=process-id	/NOIDENTIFICATION
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The SHOW SERVER command displays information about server channels.

Information such as PID, key range, state, event mask, event name and partition ID can be displayed.

The "State:" field of the SHOW SERVER command can display the following values:

Table 6–22 Key-range States

State	Meaning
wt_tr_ok	Server is waiting for routers to accept it
wt_quorum	Server is waiting for backend to be quorate
lcl_rec	Local recovery
lcl_rec_fail	Primary server waiting for access to a restart journal
lcl_rec_icpl	Getting next journal to recover from
lcl_rec_cpl	Processed all journals for local recovery
shd_rec	Shadow recovery
shd_rec_fail	Shadow server waiting for access to a restart journal
shd_rec_icpl	Shadow getting next journal to recover from
shd_rec_cpl	Processed all journals for shadow recovery
catchup	Secondary is catching up with primary
standby	Server is declared as standby
active	Server is active
pri_act	Server is active as primary shadow
sec_act	Server is active as secondary shadow
remember	Primary is running without shadow secondary

The "Flags:" field of the SHOW SERVER command can display the following values:

Table 6–23 Server Flags

FLAG	Meaning
BEC	Backend Call-out
EXA	Explicit Accept
EXP	Explicit Prepare
NCC	No Concurrent
NSB	No Standby
SHD	Shadow
SRV	Server
TRC	Router Call-out

Qualifiers

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/IDENTIFICATION=process-id
/NOIDENTIFICATION (D)
Specifies the PID of the process for which information should be displayed. The default (/NOIDENTIFICATION) displays information for all servers.

/FACILITY
/FACILITY="*" (D)
Specifies the facility name for which information should be displayed.

By default, information is displayed for all Facilities.

/FULL
/NOFULL (D)
Specifies a detailed listing of server information.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Examples

```
RTR> SHOW SERVER
Servers:

Process-id   Facility                    Channel   Flags     State
20828        RTR$DEFAULT_FACILITY        589825    SRV       active
20828        RTR$DEFAULT_FACILITY        655362    SRV       active

RTR> SHOW SERVER/FULL
Servers:

Process-id:            20828     Facility:       RTR$DEFAULT_FACILITY
Channel:              589825     Flags:                          SRV
State:                active     Low Bound:                     aaaa
High Bo                 mmmm     rcpnam:                 "LOWER_SERV"
Uents:                     0     RTR Events:                       0
Partition-Id:       16973824

Process-id:            20828     Facility:       RTR$DEFAULT_FACILITY
Channel:              655362     Flags:                          SRV
State:                active     Low Bound:                     nnnn
High Bo                 zzzz     rcpnam:                 "UPPER_SERV"
User Events:               0     RTR Events:                       0
Partition-Id:       16842753
```

SHOW TRANSACTION

Displays information about currently active transactions.

Format

SHOW TRANSACTION

Command Qualifiers	Defaults
/BACKEND	/NOBACKEND (D)
/CLUSTER	/NOCLUSTER
/FACILITY	/FACILITY="*"
/FRONTEND	/NOFRONTEND (D)
/FULL	/NOFULL
/IDENTIFICATION=process-id	/NOIDENTIFICATION
/NODE[=node-lisft]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/ROUTER	/NOROUTER (D)

Description

The SHOW TRANSACTION command displays transaction information, such the transaction ID, facility, transaction state, frontend user, start time and router node.

Qualifiers

/BACKEND
/NOBACKEND (D)
Specifies that information should be listed for transactions in a backend node. If none of /BACKEND, /FRONTEND or /ROUTER are specified, then information for all of them are displayed. If either /FRONTEND or /ROUTER are specified, then the default (/NOBACKEND) inhibits display of backend transaction information.

The "Invocation" field of SHOW TRANSACTION /BACKEND /FULL shows the following information:

Table 6–24 Transaction Invocation Types

Type	Meaning
ORIGINAL	Original transaction
REPLAY	Replayed transaction
RECOVERY	Shadow recovery transaction

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FRONTEND
/NOFRONTEND (D)
Specifies that information should be listed for transactions in a frontend node. If none of /BACKEND, /FRONTEND or /ROUTER are specified, then information for all of them are displayed. If either /BACKEND or /ROUTER are specified, then the default (/NOFRONTEND) inhibits display of frontend transaction information.

/IDENTIFICATION=process-id
/NOIDENTIFICATION (D)
Specifies the PID of the process for which information is displayed. The default (/NOIDENTIFICATION) displays information for all processes.

/FACILITY
/FACILITY="*" (D)
Specifies the facility name for which information should be displayed.

By default, information is displayed for all Facilities.

/FULL
/NOFULL (D)
Produces a detailed listing of transaction information.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in `node-list`. If `node-list` is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file `file-spec`. If `/OUTPUT` or `file-spec` is omitted then the standard or default output is used.

/ROUTER
/NOROUTER (D)
Specifies that information should be listed for transactions in a router node. If none of /BACKEND, /FRONTEND or /ROUTER are specified, then information for all of them are displayed. If either /BACKEND or /FRONTEND are specified, then the default (/NOROUTER) inhibits display of router transaction information.

The "Key-Range-State" field for SHOW TRANSACTION /ROUTER /FULL shows the following states:

Table 6–25 Key-Range States

State	Meaning
locl_rec	Local recovery in progress
shad_rec	Shadow recovery in progress

(continued on next page)

Table 6–25 (Cont.) Key-Range States

State	Meaning
active	Active non-shadowed
sec_act	Secondary active
pri_act	Primary active
pri_lone	Primary running alone
sec_chup	Secondary is catching up

Examples

```
 RTR>  SHOW TRANSACTION/BACKEND/FULL
Backend transactions:

Tid:                 e100b810,0,0,0,0,a85,83290001
Facility:       RTR$DEFAULT_FACILITY
Frontend:                    *****   FE-User:                anders.7780
State:                   RECEIVING   Start-Time:Tue Feb 21 15:53:53 1995
Router:                     bronze   Invocation:                ORIGINAL
Active-Key-Ranges:               1   Recovering-Key-Ranges:            0
Total-Tx-Enqs:                   1   Key-Range-Id:              16973824
Server-Pid:                  20828   Server-State:             RECEIVING
Journal-Node:     iron.zuo.dec.com   Journal-State:              SENDING
First-Enq:                       1   Nr-Enqs:
Nr-Replies:                      0
```

SPAWN

Allows you to execute operating system commands without leaving the RTR utility.

Format

SPAWN [operating-system-command]

Command Qualifiers	Defaults
/INPUT=file-spec	/NOINPUT
/OUTPUT=file-spec	/OUTPUT=stdout
/WAIT	/WAIT

Description

The SPAWN command allows you to execute operating system commands without leaving the RTR session. If you specify an operating system command as the parameter to SPAWN, the command is executed in the context of a spawned subprocess. For example, the command "SPAWN mail" invokes the mail utility; when you exit from mail, you return to your RTR session.

If you do not specify a parameter, the SPAWN command enters the operating system command level in the spawned subprocess. You can then enter commands; you can return to your RTR session by exiting the subprocess.

Parameters

operating-system-command
Can be any operating system command.

Qualifiers

/INPUT=file-spec
/NOINPUT (D)
Specifies an input file containing one or more shell commands to be executed by the spawned subprocess. If you specify a command and an input file (with the /INPUT qualifier), the command string is processed before the input file. Once processing of the input file is complete, the subprocess is terminated.

/OUTPUT=file-spec
/OUTPUT=stdout (D)
/NOUTPUT
Requests that the output from the SPAWN operation is written to the file-spec.

START RTR

Start RTR on one or more nodes.

Format

START RTR

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

The following qualifiers are only relevant when running on OpenVMS.

Command Qualifiers	Defaults
/ASTLM=ast-limit	Dependent on /LINKS and /PROCESSES values
/BIOLM=io-buffered	Dependent on /LINKS and /PROCESSES values
/BYTLM=buffer-limit	/BYTLM=1,000,000
/CPULM=time-limit	/CPULM=default-time-limit
/DIOLM=io-direct	/Dependent on /LINKS and /PROCESSES values
/ENQLM=enqueue-limit	/ENQLM=2000
/FILLM=file-limit	Dependent on /LINKS and /PROCESSES values
/JTQUOTA=job-table-quota	/JTQUOTA=5000
/LINKS=max-links	/LINKS=512
/PARTITIONS=max-partitions	/PARTITIONS=80
/PGFLQUOTA=page-file	Dependent on /LINKS and /PROCESSES values
/PRCLM=subprocess-limit	/PRCLM=10
/PRIORITY=priority	/PRIORITY=6
/PROCESSES=max-processes	/PROCESSES=64
/TQELM=queue-limit	/TQELM=2000
/WSDEFAULT=working-set	/WSDEFAULT=2000
/WSEXTENT=extent	/WSEXTENT=20000
/WSQUOTA=max-working-set	/WSQUOTA=10000

Description

The START RTR command starts RTR on one or more nodes.

When RTR is started on a node, a detached process called the **RTR ACP** is created. This process performs transaction and communication activities for all users of RTR running on that node.

The START RTR command must be issued before adding RTR facilities or starting application programs that use RTR.

When running on OpenVMS, the quota qualifiers affect the way the RTR ACP is created. The default values will suffice for many applications. The values for quota qualifiers take effect independently of SYSGEN settings. This means that RTR can be started with different quotas without rebooting the system. The quota qualifiers are similar to those which can be specified with the DCL RUN /DETACHED command.

If you use obsolete RTR Version 2 qualifers with the START RTR command, a warning is issued. Qualifers affected are partitions, cache_pages, and relations. Warnings are also generated if an OpenVMS qualifer is used on a non-OpenVMS platform.

Qualifiers

/ASTLM=ast-limit
/ASTLM=(max-links + max-processes)× 2 + 10 = default ast-limit (D)
Specifies the AST limit for the RTR ACP.

The value for `ast-limit` must include five for RTR ACP mailbox reads and timer scheduling and a minimum of two per DECNET logical link maintained by RTR.

For example, in a 20-node configuration, a router node needs an ASTLM of at least 45 (5 + (20 × 2)).

In systems where a lot of traffic is expected, defining additional ASTLM quota will enable lookahead I/O to be booked to channels without RTR ACP being held up in a resource wait.

The default value of `ast-limit` is automatically calculated, based on the value of /LINKS and /PROCESSES.

/BIOLM=io-buffered
/BIOLM=(max-links + max-processes)× 2 + 10 = default io-buffered (D)
Specifies the maximum number of system-buffered I/O operations that the RTR ACP can have outstanding at any one time.

The default value of `io-buffered` is automatically calculated, based on /LINKS and /PROCESSES values.

/BYTLM=buffer-limit
/BYTLM=1,000,000 (D)
Specifies the maximum amount of memory, in bytes, that the RTR ACP can use for buffered I/O operations or temporary mailbox creation.

This should be sufficiently large to account for lookahead DECNET traffic.

The default for `buffer-limit` is 1,000,000.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither `/NODE` nor `/CLUSTER` is specified then the command is executed on the nodes specified by the latest `SET ENVIRONMENT` command. If no `SET ENVIRONMENT` command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/CPULM=time-limit
/CPULM=default-time-limit (D)
Specifies the maximum amount of CPU time (in delta time) allocated to the RTR ACP. The unit of `time-limit` is ten milliseconds. When the time expires, the RTR ACP is deleted.

The default value is established at system generation time.

A `time-limit` limit of 0 indicates that CPU time is not restricted.

/DIOLM=io-direct
/DIOLM=(max-links + max-processes)× 2 + 10 = default io-direct (D)
Specifies the maximum number of direct I/O operations that the RTR ACP can have outstanding at any one time.

If you do not specify a direct I/O quota, the default value established at system generation time is used.

The default for `io-direct` is automatically calculated based on the values for /LINKS and /PROCESSES.

/ENQLM=enqueue-limit
/ENQLM=2000 (D)
Specifies the maximum number of locks that the RTR ACP can have outstanding at any one time.

The default for `enqueue-limit` is 2000.

/FILLM=file-limit
/FILLM=(max-links + max-processes) + 10 = default file-limit (D)
Specifies the maximum number of files that the RTR ACP can have open at any one time.

The default value of `file-limit` is automatically calculated, based on the values of /LINKS and /PROCESSES.

/JTQUOTA=job-table-quota
/JTQUOTA=5000 (D)
Allows you to specify a quota the job-wide logical name table for the RTR ACP.

The default for `job-table-quota` is 1024.

/LINKS=max-links
/LINKS=512 (D)
Specifies the maximum number of nodes with which this node can communicate. The default for `max-links` is 512.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in `node-list`. If `node-list` is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file `file-spec`. If `/OUTPUT` or `file-spec` is omitted then the standard or default output is used.

/PGFLQUOTA=page-file
/PGFLQUOTA=((max-links + max-processes)× 400) + 16000 =default pages (D)
Specifies the maximum number of pages allocated in the paging file for the RTR ACP.

The paging file quota is the amount of secondary storage available during execution of the RTR ACP image.

Note that a shortage of virtual memory may produce spurious error messages.

The default value of page-file is automatically calculated, based on the values of /LINKS and /PROCESSES.

/PRCLM=subprocess-limit
/PRCLM=10 (D)
Specifies the maximum number of subprocesses that the RTR ACP can create.

The default for subprocess-limit is 10.

/PRIORITY=priority
/PRIORITY=6 (D)
Requires alter priority (ALTPRI) privilege to set the priority higher than your current process.

Specifies the base priority at which the RTR ACP executes.

The priority value is a decimal number from 0 through 31, where 31 is the highest priority and 0 is the lowest. Normal priorities range from 0 through 15; real-time priorities range from 16 through 31.

Higher priorities result in faster response, but lower priorities may be more CPU efficient and give a higher overall throughput.

/PROCESSES=max-processes
/PROCESSES=64 (D)
Specifies the maximum number of processes that can use RTR on this node. The max-processes value is set to the maximum number of processes allowed by OpenVMS. The default is 64.

/TQELM=queue-limit
/TQELM=2000 (D)
Specifies the maximum number of timer queue entries that the RTR ACP can have outstanding at any one time. This number includes timer requests and scheduled wakeup requests.

The default for queue-limit is 2000.

/WSDEFAULT=working-set
/WSDEFAULT=2000 (D)
Specifies the default working set size for the image running in the RTR ACP.

working-set cannot be greater than the working set quota (specified with the /WSQUOTA qualifier).

The default for working-set is 2000.

/WSEXTENT=extent
/WSEXTENT=20,000 (D)
Specifies the maximum size to which the image running in the RTR ACP can increase its physical memory size.

The default for extent is 20,000.

/WSQUOTA=max-working-set
/WSQUOTA=10,000 (D)
Specifies the maximum size to which the image being executed in the RTR ACP process can increase its working set size.

The default for max-working-set is 10,000 pages.

Related Commands

- SHOW RTR
- STOP RTR

Examples

See Chapter 2, Starting and Setting Up RTR, for examples of how to use the START RTR command.

STOP RTR

Stop RTR on one or more nodes.

Format

STOP RTR

Command Qualifiers	Defaults
/ABORT	/NOABORT
/CLUSTER	/NOCLUSTER
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The STOP RTR command stops RTR in an orderly manner.

Alternatively, RTR can be stopped in an abrupt manner (/ABORT), and any applications using RTR are forced to exit.

Qualifiers

/ABORT
Specifying /ABORT causes RTR to stop, regardless of the state of any RTR user applications. Applications using RTR are forced to exit.

Note

This qualifier supersedes the /NOCONFIRM of earlier versions of RTR.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

Related Commands

- SHOW RTR
- START RTR

Examples

See Chapter 2, Starting and Setting Up RTR, for examples of how to use the STOP RTR command.

TRIM FACILITY

Removes nodes or roles or both from an existing facility definition.

Format

TRIM FACILITY [facility_name]

Command Qualifiers	Defaults
/BACKEND=backend-list	/NOBACKEND
/CLUSTER	/NOCLUSTER
/FRONTEND=frontend-list	/NOFRONTEND
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout
/ROUTER=router-list	/NOROUTER

Description

The TRIM FACILITY command removes nodes or roles or both from an RTR facility definition. A node can be removed from a facility or a role can be removed from a node. For example, a node having both router and frontend roles can be trimmed to have only a router role.

A TRIM FACILITY command resulting in the removal of all roles is equivalent to a DELETE FACILITY command.

A frontend role can be removed from a node with the router role as long as there is at least one other frontend defined.

A backend role can be removed from a node with the router role as long as at least one other backend is defined. A command to remove a backend role should be executed on all router nodes to avoid problems with inconsistent facility definitions.

If a router role is removed, then the node from which it is removed will discard its knowledge of other backends and frontends in the facility. If the router role is re-added to the node (using EXTEND FACILITY), this information will have to be specified again.

In order to have a consistent facility definition across the nodes of the facility (avoiding problems with attaining quorum), the command to remove a router role must be executed on all relevant nodes. This means executing the command on the node losing the router role, plus all backend and frontend nodes which know about this router.

As with CREATE or MODIFY FACILITY, nodes or roles may be specified which are superfluous. That is, you may specify backend nodes on a node which only has a frontend role, and frontend nodes may be specified on a node which only has a backend role. This permits a single RTR management command to be issued on many nodes, and each node only accepts those parts of the command which are relevant to it.

When using this command, the facility being extended may lose quorum until the affected nodes agree upon the new facility definition. During this time server applications will not be presented with any new transactions.

RTR MONITOR QUORUM displays a monitor picture which allows the quorum negotiations to be observed. You can use this after using a TRIM FACILITY command; once quorum has been re-attained, the participating nodes return to the quorate state.

For example, in a three node facility called facnam, the nodes FE and NFE have only frontend roles, and node FETRBE has frontend, router and backend roles. This facility could have been created as follows:

```
% RTR
RTR> SET ENVIRONMENT /NODE=(FE,FETRBE,NFE)
RTR> CREATE FACILITY facnam /FRONTEND=(NFE,FE,FETRBE) -
    /ROUTER=FETRBE -
    /BACKEND=FETRBE
```

The frontend node NFE can be removed from the facility as follows:

```
% RTR
RTR> SET ENVIRONMENT /NODE=(FETRBE,NFE)
RTR> TRIM FACILITY facnam /FRONTEND=NFE
```

Parameters

facility_name

Specifies the name of the facility to be trimmed.

The default value for facility_name is RTR$DEFAULT_FACILITY.

Qualifiers

/BACKEND=backend-list
/NOBACKEND (D)

Specifies the names of the nodes where backend roles for this facility are removed.

backend-list is a list of backend-nodes separated by commas. If there is more than one backend-node, then backend-list must be enclosed in parentheses.

backend-node is either the name of a node or @file-spec, where file-spec specifies a text file containing a backend-list on each line.

/CLUSTER
/NOCLUSTER (D)
Specifies that the command is executed on all the nodes in the cluster.

If neither /NODE nor /CLUSTER is specified then the command is executed on the nodes specified by the latest SET ENVIRONMENT command. If no SET ENVIRONMENT command has been entered then the command is executed only on the node where the command was issued.

Note: In environments that do not support clustering, use of the /CLUSTER qualifier will cause the relevant command to be executed on the local node only.

/FRONTEND=frontend-list

/NOFRONTEND (D)
Specifies the names of nodes where the frontend role is removed for this facility. frontend-list is a list of frontend-nodes separated by commas. If there is more than one frontend-node, then frontend-list must be enclosed in parentheses.

frontend-node is either the name of a node or @file-spec, where file-spec specifies a text file containing a frontend-list on each line.

/NODE[=node-list]
/NODE=default-node-list (D)
Specifies that the command is executed on all nodes specified in node-list. If node-list is omitted then the command is executed only on the node where the command was issued.

/OUTPUT[=file-spec]
/OUTPUT=stdout (D)
Specifies that the resulting information is written to the file file-spec. If /OUTPUT or file-spec is omitted then the standard or default output is used.

/ROUTER=router-list
/NOROUTER (D)
Specifies the names of the nodes where the router role is removed for this facility.

router-list is a list of router-nodes separated by commas. If there is more than one router-node, then router-list must be enclosed in parentheses.

router-node is either the name of a node or @file-spec, where file-spec specifies a text file containing a router-list on each line.

Related Commands

- SHOW FACILITY
- DELETE FACILITY
- CREATE FACILITY
- EXTEND FACILITY

UNREGISTER RESOURCE MANAGER (UNREGISTER RM)

The UNREGISTER RESOURCE MANAGER command unregisters an instance of a resource manager.

Format

UNREGISTER RESOURCE MANAGER [resource_name]

UNREGISTER RM [resource_name]

Command Qualifiers	Defaults
/CLUSTER	/NOCLUSTER
/NODE[=node-list]	/NODE=default-node-list
/OUTPUT[=file-spec]	/OUTPUT=stdout

Description

The UNREGISTER RESOURCE MANAGER command unregisters an instance of a resource manager. The command succeeds only when no facility references it. To unregister a resource manager instance, you must first delete all facilities that reference it. To see what facilities reference a specific resource manager, use the command SHOW RM/FULL. Refer to Appendix C, XA Support for support information about XA.

Parameters

resource_name
Specifies the name of the resource to be unregistered.

Resource names can contain up to 32 characters. This argument is required.

Related Commands

- REGISTER RESOURCE MANAGER
- SHOW RESOURCE MANAGER

Examples

```
UNREGISTER RM rmi_1
```

A

Creating Monitor Pictures

The standard monitor pictures provided with RTR (described in Chapter 5) are sufficient for most needs. You may also create your own monitor pictures to suit particular needs. This appendix tells you how to do this.

The RTR monitor utility provides a means to continuously display the status of RTR and the applications using it. The information displayed is composed of named data items which are continuously updated by RTR. The data items can be displayed in various formats, and combined using simple arithmetic operators and constants.

Note that in earlier versions of RTR, the data items were known as counters, and only contained numeric information. The name has been changed to indicate that string data may also be retrieved.

All RTR data items are associated with an information class. When entering a data item in a display command, it may be prefixed with a string to indicate which class it belongs to. This can speed up the processing of monitor commands. Information classes are shown in Table A–1.

Table A–1 Information Classes

Information Class	String
per node	"rtr"
per facility	"fac"
per link to a node	"lnk"
per client process	"cli"
per server process	"srv"
per application process	"prc"
per partition on a router	"rpt"
per partition on a backend	"bpt"
per key segment	"ksg"
per transaction on a frontend	"ftx"
per transaction on a router	"rtx"
per transaction on a backend	"btx"

The monitor is invoked with the RTR MONITOR command. RTR monitor displays a monitor picture that is periodically updated. See Section 6.2 for the full syntax of the MONITOR command.

A monitor picture contains elements that are either text (such as labels and titles) or variables derived from data items. Monitor pictures can be defined either interactively at the RTR> prompt or defined in a file called a monitor file.

The commands used to define and display monitor pictures are:

CLEAR
DISPLAY BAR
DISPLAY NUMERIC
DISPLAY STRING
DISPLAY SYMBOLIC
DISPLAY TEXT
MONITOR
SCROLL
SHOW DISPLAY

These commands are described in the following sections.

A.1 Interactive Definition of a Monitor Picture

Example A–1 shows a monitor picture being defined in an interactive RTR session.

Example A–1 Interactive Picture Definition

```
$ RTR
RTR> CLEAR /ALL  1
RTR> DISPLAY TEXT /X=25 /Y=1 "THE TEST PICTURE AT $TIME" 2
RTR> DISPLAY NUMERIC RTR_SOME_DATA_ITEM -  3
_RTR>        /X=1 /Y=3 /LABEL ="SOME DATA ITEM:   "
RTR> DISPLAY NUMERIC RTR_ANOTHER_DATA_ITEM -  4
_RTR>        /X=1 /Y=4 /LABEL ="OTHER DATA ITEM: "
RTR> MONITOR  5
RTR> CLEAR /X=25 /Y=1  6
RTR> DISPLAY TEXT /X=25 /Y=1 "THE NEW PICTURE AT $TIME" 7
RTR> SHOW DISPLAY /OUTPUT=MYPICTURE.MON  8
```

1 Any items from previously displayed pictures are cleared.

2 A title is displayed on the first line. The symbol $TIME is substituted by the current system time when the picture is displayed. (See Section A.2, Substitution Symbols).

3 The data item called RTR_SOME_DATA_ITEM is displayed on the third line.

4 The data item called RTR_ANOTHER_DATA_ITEM is displayed on the next line. Note that if /X and /Y are omitted, the item is displayed on the next free line and in the same column as the previous item.

5 The monitor picture is displayed on the screen. Figure A–1 shows what would appear on the screen at this point.

6 The title is removed from the picture definition.

7 A new title is added. This technique can also be used to make small alterations to standard monitor pictures.

8 The definition of the picture is saved in the text file MYPICTURE.MON. The saved picture can be redisplayed by entering MONITOR MYPICTURE.

Figure A–1 Interactively Defined Monitor Picture

```
                        THE TEST PICTURE AT 11:49:24

SOME DATA ITEM:   0
OTHER DATA ITEM:  0
```

Caution

Because monitor file definitions depend on the internal structure and data items of RTR, they may need to be changed for future versions of RTR.

A.2 Substitution Symbols

The text and labels in the DISPLAY commands that are used to define monitor pictures can contain symbols which are substituted when the picture is displayed. Table A–2 contains a list of these symbols.

Table A–2 Substitution symbols

Symbol	Description
$TIME	Current time
$DATE	Current date
$NODE_NAME	Name of node where values were measured
$LINK_NAME	Name of link for which values were measured
$FACILITY_NAME	Name of facility for which value was measured
$PROCESS_ID	ID of process for which value was measured
$PROCESS_NAME	Name of process for which value was measured
$IMAGE_NAME	Image file name of process for which value was measured
$FULL_IMAGE_NAME	The full file specification of the image for the process for which values were measured

The field width for these predefined symbols can be specified by appending a colon (":") and then the required width. For example, `$TIME` displays the current system time in the form "10:34:03", `$TIME:5` displays the first five characters of the system time, in this case, "10:34". If the field width specified is larger than the field, then the data is left justified and space filled to the required width.

A.3 Arithmetic Expressions and Operators

When using data items and constants as parameters for display commands, or as Booleans in the BELL, BLANK, BLINK, BOLD, SELECT, REVERSE and UNDERLINE qualifiers, you may use the operators shown in Table A–3.

Table A–3 Arithmetic Operators in Display Commands

Characters	Meaning
-	Minus
+	Plus
*	Multiply
/	Divide
&	Logical AND
|	Logical OR
<=	Less than or equal
>=	Greater than or equal
!=	Not equal
<>	Not equal
<	Less than
>	Greater than
=	Equal

Example A–2 shows how these operators are used.

Example A–2 Arithmetic Operators Examples

```
RTR> DISPLAY NUMERIC RTR_SOME_DATA_ITEM+2      1
RTR> MONITOR

RTR> DISPLAY NUMERIC RTR_SOME_DATA_ITEM
_RTR> /SELECT="RTR_SOME_OTHER_DATA_ITEM>42"    2
RTR> MONITOR

RTR> DISPLAY NUMERIC RTR_SOME_DATA_ITEM
_RTR> /SELECT="RTR_SOME_OTHER_DATA_ITEM>42" -
_RTR> /BOLD="RTR_SOME_OTHER_DATA_ITEM>84"  -
_RTR> /UNDERLINE="RTR_SOME_OTHER_DATA_ITEM>=1952"  3
RTR> MONITOR

RTR> DISPLAY SYMBOLIC RTR_SOME_DATA_ITEM "zero","one" -
_RTR> "two","three","four,"five","six"/BLINK        4
```

1 Two is added to the value of RTR_SOME_DATA_ITEM before displaying it.

2 RTR_SOME_DATA ITEM is only displayed if RTR_SOME_OTHER_DATA_ ITEM is greater than 42.

3 Same as the one above, but displayed in bold if RTR_SOME_OTHER_DATA_ ITEM is greater than 84, and underlined if its greater than 1952.

4 The value of RTR_SOME_DATA_ITEM is displayed in text form if it is less than six, otherwise it is displayed as a numeric.

Aggregation of Data Items

DISPLAY commands which select multiple instances of a data item (for example, multiple instances of a process counter) can use the following keywords to control the way the items are aggregated.

_MIN—select the data item instance with the lowest value
_MAX—select the data item instance with the higheest value

By default, data items are totaled unless the /AVERAGE=( ) qualifier is specified.

The following example from SYSTEM.MON shows the use of the _MIN keyword:

```
! Warn if any messages have been pending for more than 30 seconds
DISPLAY TEXT    /X=32    /Y=17 -
        "X" -
        /BOLD -
        /SELECT="...
       ((rtr:rtr_current_time - _MIN(prc:rtr_mt_received_time)) <= 30)"

DISPLAY TEXT    /X=40    /Y=17 -
        "X" -
        /BOLD /BLINK -
        /SELECT="...
       ((rtr:rtr_current_time - _MIN(prc:rtr_mt_received_time)) > 30)"
```

Some output has been omitted for clarity.

B

Server Shadowing and Recovery

RTR **shadowing** gives you the ability to recover from a site disaster without the need for special coding within your application program. This appendix is an introduction to RTR shadowing.

A server for a database partition is said to be shadowed when two copies of the same server perform identical actions on identical database copies on separate nodes. It is possible to declare servers to be members of a pair of shadow sites for any particular RTR key-range. Shadowed sites can either be two nodes within a single cluster, or can be two separate clusters.

Note that concurrent servers handle `similar` transactions, (that is, in the same key-range but not the same transactions). Standby servers do not handle transactions at all (for the given key-range) and shadow servers handle the `same` transactions.

B.1 Primary and Secondary Roles

There is a concept of primary and secondary role for the shadow server pair, although in most cases this is transparent to the user when the processing is the same on both sites.

Initial role assignment is arbitrary, in that the first server of a shadow pair to start is given the primary role, and the second the secondary. The assigned roles may change, as servers come and go. Roles are required, since RTR needs to determine the voting order on the primary site before the transaction is presented to the secondary site.

B.2 Automatic Features

Shadow sites each have an identical copy of the customer's database.

Transactions are sent by RTR to `both` sites, and RTR ensures that they are processed by the servers in equivalent order on each of these sites, so that both copies of the customer database remain up-to-date.

A transaction is sent to the secondary site only after the primary has accepted it, or if the primary fails before being asked to vote.

RTR suppresses replies and broadcasts issued by the secondary shadow server.

B.2.1 Shadow Events

RTR provides the following shadowing events:

- RTR_EVTNUM_SRPRIMARY - Server is in primary mode
- RTR_EVTNUM_SRSTANDBY - Server is in standby mode
- RTR_EVTNUM_SRSECONDARY - Server is in secondary mode
- RTR_EVTNUM_SRSHADOWLOST - Server has lost its shadow partner

- RTR_EVTNUM_SRSHADOWGAIN - Server has gained its shadow partner
- RTR_EVTNUM_SRRECOVERCMPL - Server has completed recovery

The shadow events are delivered with no special status and no data. They are delivered only to the server(s) whose state has changed.

A server receives RTR_EVTNUM_SRPRIMARY under the following circumstances:

- On initial startup if servers for the key-range are not already running on other nodes.
- If the server had previously been standby and the previous primary has failed.
- If the server had previously been shadow secondary and the previous primary has failed.

A server receives RTR_EVTNUM_SRSTANDBY when it starts up and servers already exist for the same key-range on another node in the `same cluster`.

A server receives RTR_EVTNUM_SRSECONDARY when it starts up and a shadow primary set of servers exist elsewhere.

A server receives RTR_EVTNUM_SRSHADOWLOST if it is running as primary and the secondary goes away.

A server receives RTR_EVTNUM_SRSHADOWGAIN if it is running as primary and a secondary node starts up.

A server receives RTR_EVTNUM_SRRECOVERCMPL when it has finished doing recovery operations, and is hence ready to start processing new transactions.

B.3 The RTR Journal System

The RTR journal is used for two purposes:

- RTR stores data about a transaction so that if the server processing a transaction fails for any reason, another server can transparently continue from where the previous server failed (if it is able to access the same database and journal).
- RTR also stores data about transactions when a shadow site is known to be missing. In this case, RTR stores all transaction data which can result in a database update. The RTR journal stores this data until the secondary shadow site is available again. RTR then transparently replays this data to the shadow site, after which the data is deleted from the journal.

The amount of space required for the journal depends upon:

the size of the messages in a transaction,
the number of messages in the transaction,
the rate of generation of transactions,
the maximum time a shadow site can be out of commission.

The /MAXIMUM_BLOCKS qualifier on the CREATE JOURNAL command controls how large a journal may become. The /MAXIMUM_BLOCKS qualifier defines the maximum number of blocks which the journal is allowed to occupy on any one disk. Note that RTR does not check if this amount of space is actually available, as the disk space specified by /MAXIMUM_BLOCKS is used only on demand by RTR when insufficient space is available in the space allocated by the /BLOCKS qualifier.

The number of blocks specified by the /BLOCKS qualifier specifies the size of the journal that RTR attempts to keep to. (The actual number of blocks used may vary, depending upon the load on RTR.)

The command MODIFY JOURNAL also accepts the /BLOCKS and /MAXIMUM_ BLOCKS qualifiers.

Journal file extension occurs on demand when RTR detects that a "write to journal" would otherwise fail due to lack of space. Journal file truncation takes place automatically when blocks are freed.

Refer to MODIFY JOURNAL for the syntax description of the MODIFY JOURNAL command.

```
RTR> show journal/files/full

RTR journal:-

Disk:   /dev/rz3a       Blocks:     2500 Allocated: 1253 Maximum: 3500
File:   //rtrjnl/anders/BRONZE.J00

RTR>
```

B.4 Shadow Site Failure and Journaling

If a shadow-site fails, RTR allows transactions to continue to be processed on the remaining site. The intermediate transactions processed by the remaining server or servers are retained by RTR; when the failed site restarts these transactions are sent to this site as part of a shadow-recovery operation, thus bringing the failed site back up-to-date.

Since the transactions are stored in the RTR journal, it must be created with enough disk space in reserve to store data for the longest expected outage. It can be calculated using:

```
( Nr. transaction messages per second
  multiplied by ( transaction message length + 70 )
  multiplied by seconds of outage
) + 5% file overhead.
```

The overhead required when calculating journal size comes from internal journal data (block stamping) of approximately 3%. In addition, there is internal transaction data per (client to server) transactional message, and some further data per transaction (concerning voting and transaction completion).

Also, note that RTR prevents further transactional data being written to the journal when it is nearly full, but continues to allow deletes from the journal (deletes also cause data to be written to the journal). Ten segments are held in reserve for storing information about deleted transactions even when RTR cannot accept further transactions because the journal is full.

Warning

If the journal disk becomes full, transactions are aborted until the shadow partner re-starts and empties the journal of transactions to be replayed.

B.4.1 Maximum Journal Size

The current maxima for the size of a journal are:

Number of blocks per disk: 524288

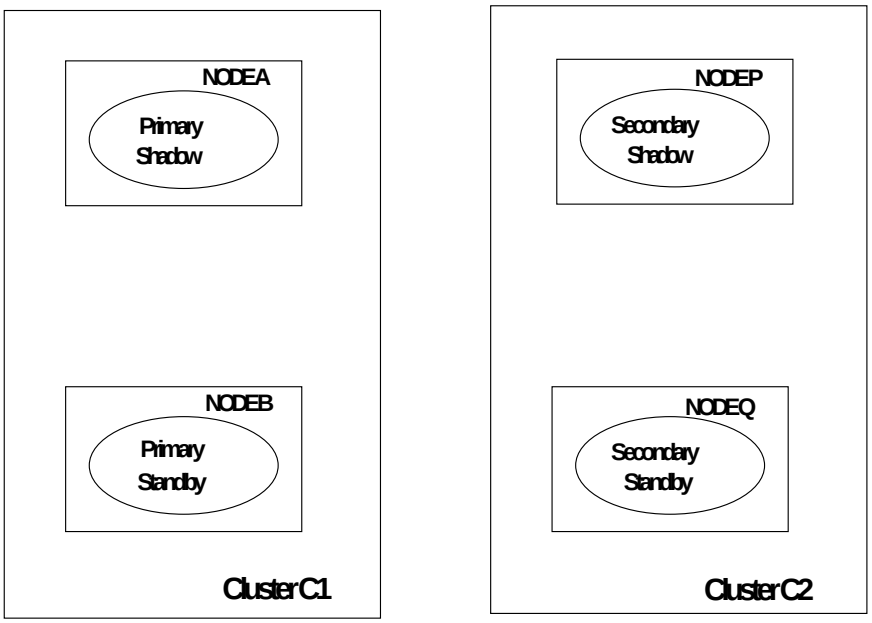
(This is `max_segments_per_disk * disk_blocks_per_segment`, or 16384 times 32.)
Number of disks per journal: 16.

B.5 Standby for Shadows

Shadowed sites can either be two nodes within a single cluster, or can be two separate clusters. In the second case it is also possible to configure standby servers on another node of each of these clusters, so that failure of a single node within one of the shadow sites does not stop the shadow site from functioning. Multiple concurrent copies of the server processes are allowed on each site.

This implies that up to four nodes can be configured to serve a particular key range. (see Figure B–1).

Figure B–1 Four Node Shadow/Standby Configuration



B.6 Performance

The performance of a shadow pair compares with a transaction which spans two nodes, with the addition of one extra protocol message which is required to ensure that the transactions are presented in the equivalent order.

Note that RTR does not have to wait for the secondary shadow server to complete its processing. It only needs to know that the primary has committed the transaction and that the journal file of the secondary shadow server contains the final vote status.

The two partners in a shadow pair should be connected with sufficient bandwidth to cater for the possibly large amounts of data which may need to be transferred during a shadow catchup operation.

B.7 Shadows in Action

The first node on which a shadow server for a particular key-range starts is arbitrarily designated by RTR to be the primary site for that key-range.

Initially RTR searches the journals of other backend sites to find any recoverable transactions left over from a previous invocation of the server. Once these have been processed (or RTR is able to determine that no such transactions exist), the server becomes active and available to handle new transactions sent by clients.

While no other server site for this key-range is available the server runs in REMEMBER mode, and RTR saves transactions processed on this site in the RTR journal (together with the order in which they should be committed), so that when the other site server(s) start they can be sent to this site.

When a server starts on a second site it begins processing the transactions saved in the primary site's journal. These are deleted from the journal as they are processed. When the second site server(s) have caught up, the second site enters SECONDARY state and the original site servers enter ACTIVE state. In this mode, new transactions are sent to both sites in parallel. They are executed first on the primary site, and shortly afterwards on the secondary site in equivalent order. The primary site commits transactions as soon as it knows that the secondary has hardened (i.e. written to the journal) the order in which the transaction is to be committed.

In the event of a failure at this point the remaining site executes a short tidy-up operation, and as soon as it has done this and determined that the other site is really down, it reverts to the REMEMBER state and carries on processing new transactions autonomously, saving the transaction information in its journal for when the other site restarts.

The execution order is determined for transactions issued to concurrent servers on a particular node by recording the order in which the individual servers issue `rtr_accept_tx( )` calls. RTR knows that at the time a correctly written server application calls `rtr_accept_tx( )`, it has already accessed (and hence locked) any database records it uses, and that it will release these records after RTR causes the `rtr_accept_tx( )` call to complete. Any conflicting transaction would not be able to issue `rtr_accept_tx( )` concurrently. Hence a correct serialisation order for issuing the transactions on the shadow site can be determined.

B.8 Application Considerations

Although applications need not be directly concerned about Shadowing matters, certain points must be taken into consideration when implementing performance boosting optimizations:

- Anything specific to the executing node should not be stored in the database, since this would lead to diverging copies.

- Any physical reference to the transaction which is unique to the executing server, e.g. Channel Id, system time, DB-key, etc., should not be passed back to the client for future references within its subsequent messages, as this could lead to inconsistent handling when a different server is involved in shadow operations.

  This consideration is also valid for recovery of non-shadowed servers.

B.9 Server States

The current state of a server can be examined as follows:

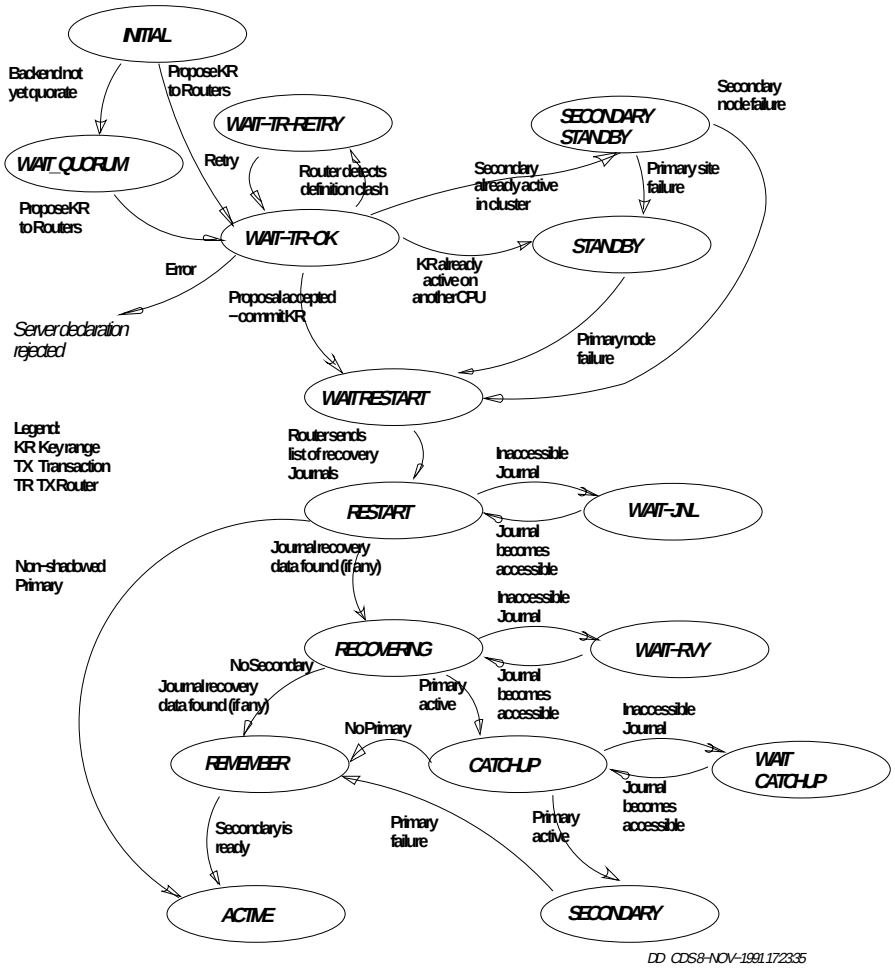
```
RTR> show server/full
Servers:

Process-id:                  13340     Facility:       RTR$DEFAULT_FACILITY
Channel:                    131073     Flags:                          SRV
State:                      active     Low Bound:
High Bound:                 87  13     rcpnam:        "RTR$DEFAULT_CHANNEL"
User Events:                     0     RTR Events:                        0
Partition-Id:            16777216

Process-id:                  13340     Facility:       RTR$DEFAULT_FACILITY
Channel:                    196610     Flags:                          SRV
State:                      active     Low Bound:                  88  13
High Bound:                    0f'     rcpnam:                    "CHAN2"
User Events:                     0     RTR Events:                        0
Partition-Id:            16777217
```

Figure B–2 gives an overview of the server state changes which appear in the "State:" field

Figure B–2 Server States



B.10 Client States

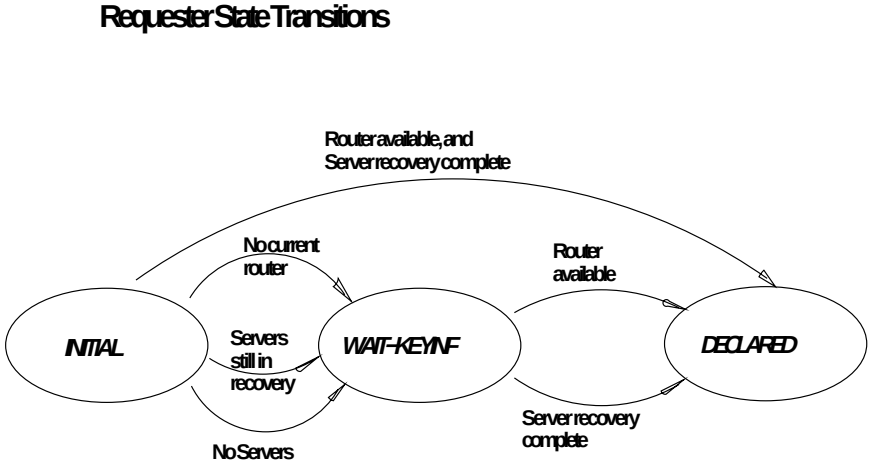
The current state of a client process can be examined as follows:

```
RTR> show client/full
Clients:

Process-id:                 13340     Facility:       RTR$DEFAULT_FACILITY
Channel:                   458755     Flags:                          CLI
State:                   declared     rcpnam:                     "CHAN3"
User Events:                  255     RTR Events:                       0
```

Figure B–3 describes the client state changes which appear in the "State:" field

Figure B–3 Client States



B.11 Partition States

The current state of a key-range partition can be examined using the SHOW PARTITION/FULL command for the routers and the backends:

```
RTR> show partition/router/full

Facility:       RTR$DEFAULT_FACILITY     State:                        ACTIVE
Low Bound:                         0     High Bound:              4294967295
Failover policy:                                             fail_to_standby
Backends:                                                              depth
States:                                                               active
Primary Main:                   depth     Shadow Main:
```

Backend partitions:

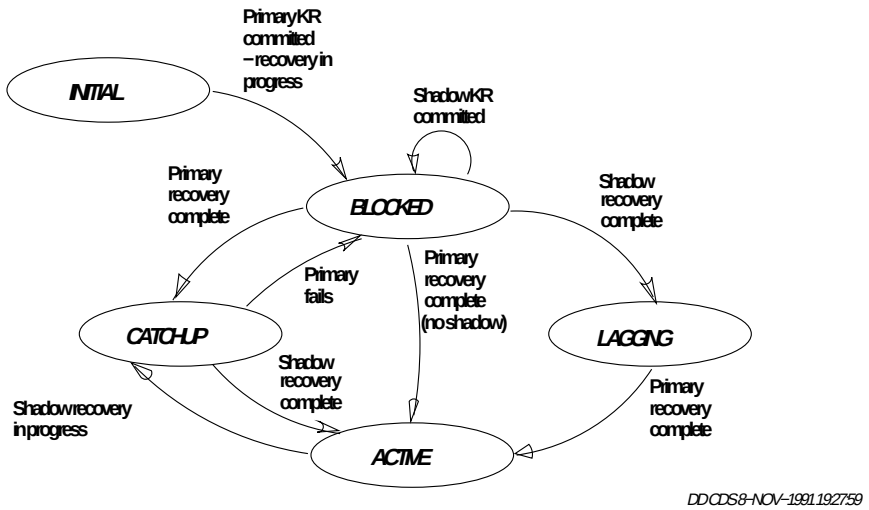
```
RTR> show partition/backend/full

Partition name:                                RTR$DEFAULT_PARTITION_16777217
Facility:       RTR$DEFAULT_FACILITY     State:                        active
Low Bound:                     "aaaa"    High Bound:                    "mmmm"
Active Servers:                    0     Free Servers:                       1
Transaction presentation:     active     Last Rcvy BE:
Txns Active:                       0     Txns Rcvrd:                         0
Failover policy:     fail_to_standby     Key range ID:               16777217

Partition name:                                RTR$DEFAULT_PARTITION_16777218
Facility:       RTR$DEFAULT_FACILITY     State:                        active
Low Bound:                     "nnnn"    High Bound:                    "zzzz"
Active Servers:                    0     Free Servers:                       1
Transaction presentation:     active     Last Rcvy BE:
Txns Active:                       0     Txns Rcvrd:                         0
Failover policy:     fail_to_standby     Key range ID:               16777218
```

Figure B–4 describes the partition state changes which appear in the "State:" field

Figure B–4 Router Partition States



C

XA Support

This appendix explains how RTR may be used with a **X/OPEN Distributed Transaction Processing (DTP)** conformant **Resource Manager**.

C.1 Introduction

The X/OPEN Distributed Transaction Processing (DTP) architecture defines a standard interface that lets application programs share resources provided by resource managers. The **XA** interface uses the two-phase commit protocol to commit transactions, and is a system-level bidirectional interface between the transaction manager **(TM)** and the resource manager **(RM)**. In the RTR environment, RTR is the transaction manager and database software such as ORACLE8 is the resource manager.

Without XA, an RTR application must deal with replayed transactions after server recovery delivered with `rtr_mt_msg1_uncertain`; the application has to check if the traansaction has been committed to the database. With XA, the application does not need to be concerned with this problem.

The XA library is an external interface that enables a transaction manager to coordinate global transactions. These can include opening a resource manager, starting a transaction, rolling back a transaction, preparing and committing a transaction, and closing a resource manager. With XA, RTR can connect directly to a resource manager such as ORACLE8.

C.1.1 MONITOR XA

Monitors the internal status of XA interface activities. The command displays counters containing information such as the number of XA calls, call status (success or failure), and the number of read-only transactions. Provides counts for the open, close, start, end, prepare, commit, rollback, and recovery commands.

Command Syntax: MONITOR XA

C.1.2 New Qualifier to CREATE FACILITY Command

The CREATE FACILITY command has a new qualifier that references the defined resource manager. All resource managers that will be accessed by a facility must be specified. All transactions to these resource managers are recovered when the facility is created.

Command Syntax:

```
CREATE FACILITY facility_name /frontend=... /backend=... /router=...
                /resource_manager=(RM_1, RM_2)
```

where:

facility_name = user-specified facility name /resource_manager = the name of those RMs this facility references

For example:

```
CREATE FACILITY test /front=N1 /back=N2/router=N3
/resource_manager=(employ_rm,payroll_rm)
```

C.1.3 Modified RTR API

An application written to use RTR with XA does not need all existing RTR application programming interface statements. In particular, there is no longer a need to include code to handle `rtr_mt_msg1_uncertain` messages.

C.1.4 RTR Open Channel

RTR relies on the server application to specify which RM it is attached to using the open channel API. The server must submit an RM instance name with open string to the RTR open channel API before accessing the desired database. Both the RM instance name and the database name (RM name) contained in open string must be the same as that registered earlier using the REGISTER RM command. This information is now stored in the RTR key segment structure. A new RTR_F_OPE_XA_MANAGED flag is created to associate the channel with the XA interface. For example:

```
typedef struct {
        rtr_keyseg_type_t ks_type; [RM_NAME_TYPE]
        rtr_keylen_t    ks_length;    [N/A]
        rtr_keylen_t    ks_offset;    [N/A]
        rtr_pointer_t   ks_lo_bound;  [RM_NAME]
        rtr_pointer_t   ks_hi_bound;  [OPEN_STRING]
} rtr_keyseg_t;
```

C.2 Microsoft DTC Support

RTR for Windows NT is interoperable with Microsoft's Distributed Transaction Controller (DTC). DTC is supported via the RTR XA software architecture. That is, with the X/Open XA protocol, RTR users can develop application programs to update MS SQL Server databases, MSMQ, or other Microsoft resource managers under the control of a true distributed transaction.

This is possible because RTR (as a distributed transaction manager) is able to directly communicate with MS DTC to manage a transaction or perform a recovery via the XA protocol. For each standard XA call received from RTR, MS DTC will translate it into a corresponding OLE transaction call that SQL Server or MSMQ can use to update databases.

D

RTR Utility Error Messages

This appendix describes the various error messages that can be returned by the RTR utility.

%RTR-F-ABKEYW, Ambiguous qualifier or keyword - supply more characters

Explanation: Too few characters were used to truncate a keyword or qualifier name to make the keyword or qualifier name unique.

%RTR-F-ABVERB, Ambiguous command verb - supply more characters

Explanation: Too few characters were used to truncate a command name to make the command name unique.

%RTR-E-ACCTOOBIG, ACCESS string is too long

Explanation: The string supplied with the "/ACCESS" qualifier on the "open channel" command was too long.

%RTR-F-ACPINSRES, The RTR ACP has insufficient resources: details in the rtr log file

Explanation: The RTR ACP was unable to perform an operation due to an unusual condition. This is most probably a resource issue, e.g. when the ACP cannot create a shared memory segment.

The RTR log file for contains more details of the problem.

%RTR-E-ACPNOTVIA, RTR ACP is no longer a viable entity, restart RTR

Explanation: Indicates that the RTR ACP process has terminated unexpectedly.

%RTR-I-ALRDYINSTATE, Partition is already in the desired state

Explanation: Returned following an attempt to change the state of shadowing for a partition when the partition was already in the desired state.

%RTR-E-AMBIGDISP, Ambiguous monitor file name, [A]

Explanation: The filename [A] could refer to more than one monitor file. Please supply more characters.

%RTR-F-AMBROUNAM, Ambiguous API routine name for CALL - supply more characters

Explanation: The parameter for the CALL command is the name (or part of a name) of an RTR API routine. This allows the user to type, e.g. "rtr call accept" instead of "rtr call rtr_accept_tx". This message is issued if the user has specified a part of an API routine name that matches more than one routine.

%RTR-F-BADDSKWRI, Unable to create/extend a journal file

Explanation: An attempt to create or extend a journal file on disk failed. Check that the disk(s) you are using for journals have sufficient free space.

%RTR-E-BADOP, Unable to complete operation @[A] line [A]

Explanation: Processing definition incomplete or undefined - report occurrence together with supporting information on current command to RTR Engineering.

%RTR-F-BADOUTFIL, Cannot open file specified with /OUTPUT

Explanation: The file specified with the /OUTPUT qualifier cannot be opened.

%RTR-E-BADPRTSTATE, Disallowed attempt to make an illegal or undefined partition state transition

Explanation: Returned following an attempt to make an illegal or undefined partition state transition.

%RTR-E-BADRTRINS, RTR is not correctly installed

Explanation: RTR is not correctly installed. Refer to the RTR Installation Guide for details of how to install RTR.

%RTR-E-BENOTALL032, Not all back-ends are at the minimum required version of V3.2

Explanation: Cannot perform the requested action since not all routers are at a minimum version of V3.2

%RTR-E-CANTSTOP, RTR could not be stopped

Explanation: Indicates that RTR cannot be stopped under present circumstances.

%RTR-E-CHAALROPE, Channel [A] is already open in this window

Explanation: An RTR channel of this name is already open in this window.

%RTR-F-CHANOTOPE, Channel not opened

Explanation: Channel was not opened. Check channels using the "SHOW CHANNEL" command.

%RTR-E-CHNALRDEC, Channel [A] is already declared

Explanation: The channel specified with the "/CHANNEL" qualifier on a "call rtr_open_channel" command has already been declared.

%RTR-E-CHNOTACTIVE, Channel does not have active transaction running

Explanation: No transaction currently active on channel. Occurs only in the V2 command environment, and is retained for compatibility with previous versions of RTR.

%RTR-E-CLASSREQ, At least one data-class definition required

Explanation: At least one data-class definition required in call to rtr_ request_info.

%RTR-E-CLOSEPEND, Send failed due to close pending on channel - call rtr_ receive_message

Explanation: Sending of data to the ACP has been aborted due to the presence of an undelivered mt_closed message on the channel. The application may retrieve the reason for the channel closure, by calling the receive_message verb to receive the mt_closed message.

%RTR-I-CMDIGNORE, Command ignored for defined facility role

Explanation: Indicates that the command is ignored on the executing node since it has no significance for the role defined.

%RTR-I-CMDNOTWRK, [A]-command not implemented

Explanation: This command is not currently implemented.

%RTR-E-CMDRESDEV, Command reserved to RTR development

Explanation: An unsupported command was issued.

%RTR-E-CMDTOOLON, Command too long

Explanation: Command was longer than 256 characters.

%RTR-E-CNTCRJOU, Cannot create journal directory

Explanation: Cannot create journal directory.

%RTR-S-COMARESEN, Commands sent by default to node [A]

Explanation: Displays the default nodes for command execution after issuing a "SET ENVIRONMENT" or "SHOW ENVIRONMENT" command.

%RTR-E-COMNOTFOU, Command not found [A], use RECALL/ALL

Explanation: Indicates that the command [A] requested with "RECALL" was not in the recall buffer.

%RTR-E-COMNUMMUS, Command number must be between 1 and [A]

Explanation: Indicates that the command number requested with "RECALL" was not in the allowed range (1 to [A]).

%RTR-F-CONFLICT, Illegal combination of command elements - check documentation
n [A]

Explanation: Two or more keywords, qualifiers or parameters that cannot be used in in combination have been in the same command line.

%RTR-I-DEQDATA, Received data ([A] bytes) [A]

Explanation: Displays the dequeued data [B] and its length in bytes [A].

%RTR-F-DFSDISK, Disk is served by DFS

Explanation: An attempt was made to create a journal on a disk served by DFS. RTR does not support journals on DFS supported disks.

%RTR-I-DISABMOD, [A] mode disabled

Explanation: Displays the name [A] of the mode that was disabled after issuing a "SET MODE" command.

%RTR-S-DISITMCLR, [A] display item(s) cleared

Explanation: Indicates how many display items [A] were successfully cleared after issuing a CLEAR DISPLAY command.

%RTR-W-DISKALL, Disk is not available to RTR

Explanation: An attempt was made to create a journal on a disk which is allocated to a different process.

%RTR-W-DISKMNTVER, Disk is currently under mount verification

Explanation: An attempt was made to create a journal on a disk which is in mount verification. Try later.

%RTR-W-DISKMOUFOR, Disk is mounted foreign

Explanation: An attempt was made to create a journal on a disk which is mounted foreign. Please check disk for proper mount status.

%RTR-W-DISKNOTMOU, Disk is not mounted

Explanation: An attempt was made to create a journal on a disk which is not mounted. Please check disk for proper mount status.

%RTR-W-DISKSSM, Disk is member of shadow set

Explanation: An attempt was made to create a journal on a disk which is a member of a shadow set. RTR cannot locate journal on individual shadow set members.

%RTR-W-DISKSWL, Disk is software write locked

Explanation: An attempt was made to create a journal on a disk which is software write locked.

%RTR-E-DSKNOTSET, Specified disk not part of the journal disk set

Explanation: A disk was specified as part of a MODIFY JOURNAL command which was not part of the original disk set specified in the CREATE JOURNAL command.

%RTR-E-DTXNOSUCHRM, There is no such RM registered

Explanation: There is no such RM registered.

%RTR-W-DTXREADONLY, The transaction branch was read only and has been committed

Explanation: The RM will simply return an warning indicating the transaction branch was read-only and has been committed already.

%RTR-E-DTXRMBUSY, DTX RM is still in use by RTR

Explanation: the DTX RM is still referenced by at least one RTR facility or open channel.

%RTR-E-DTXRMEXISTS, The DTX RM has already been registered

Explanation: The RM has already been registered.

%RTR-E-DTXTOOMANYRMS, Too many RMs or instances of an RM have been registered

Explanation: The RTR ACP has registered too many (> 16) RM instances.

%RTR-F-DUPJOUFIL, Duplicate RTR journal file found - remove duplicate file or CREATE JOURNAL /SUPERSEDE

Explanation: A duplicate RTR journal file has been found. This status may be returned by the CREATE FACILITY and SHOW JOURNAL commands.

Probable cause:

System management error. A user has copied a journal file, or a disk containing a journal file. RTR can now see both the original and the copy and does not know which to use.

Corrective action, either:

(a) Check the log for the relevant filenames, and delete or move the duplicate journal file, or

(b) Reissue the CREATE JOURNAL /SUPERSEDE command (in this case any recovery information in the old journal is lost).

%RTR-E-DUPLPARTNAME, Duplicate partition name

Explanation: Duplicate partition argument.

%RTR-E-DUPLRMNAME, Duplicate rm partition name

Explanation: Duplicate partition argument.

%RTR-E-DUPNODNAM, Duplicate node name, [A]

Explanation: The node name list specified with the "/FRONTEND", "/ROUTER" or "/BACKEND" qualifiers on an "CREATE FACILITY" command contained the node name [A] more than once.

%RTR-E-EMPTYREQ, Request_info list is empty ...

Explanation: Empty list supplied to rtr_request_info.

%RTR-I-ENABMOD, [A] mode enabled

Explanation: Displays the name [A] of the mode that was enabled after issuing a "SET MODE" command.

%RTR-E-ENODNANAM, DECnet definition required, but not found for node [A]

Explanation: A node name was specified in a context that required a successful lookup for the DECnet address, but none was available. This can occur when DECnet is explicitly specified through use of a node name prefix ("dna."), or when DECnet is the only enabled network transport. Check the network node name database on the local node, and the network name server (DNS).

%RTR-E-ENOIPNAM, IP definition required, but no host definition found for [A]

Explanation: A host name was specified in a context that required a successful lookup for the IP address, but none was available. This can occur when TCP is explicitly specified through use of a node name prefix ("tcp."), or when IP networking is the only enabled network transport. Check the host name database on the local node, and the network name server (DNS).

%RTR-F-ENOTRANSPORTS, No network transports available

Explanation: No network transport are available. Most likely cause is the manipulation of the transport protocols allowed to RTR.

%RTR-F-ERRACCDIR, Directory [A], cannot be accessed or opened

Explanation: A directory cannot be accessed or opened.

%RTR-E-ERRACCFIL, Error accessing file [A]

Explanation: Displays the name [A] of a file that the RTR utility was unable to access.

%RTR-E-ERRACCMBX, Error accessing mailbox

Explanation: An error occurred whilst accessing a mailbox. The subsequent message gives more details.

%RTR-E-ERRACCNOD, Error accessing node [A]

Explanation: An error occurred whilst accessing node [A]. If you were using the /NODE or /CLUSTER qualifier to issue a remote command please check that you are able to execute simple non-Rtr remote shell or DECnet commands on the remote node. Rtr remote commands will not work unless remote shell software is installed and proxy and rhost settings are correctly configured. Also check that both nodes have the same RTR_PREF_PROT value on platforms that can use both DECnet and TCP/IP.

%RTR-E-ERRACCTAB, Error accessing tables

Explanation: Indicates that the configuration tables could not be accessed. The subsequent message gives more details.

%RTR-E-ERRCREMBX, Error creating mailbox

Explanation: An error occurred whilst creating a mailbox. The subsequent message gives more details.

%RTR-E-ERRDELMBX, Error deleting mailbox

Explanation: An error occurred whilst deleting a mailbox. The subsequent message gives more details.

%RTR-E-ERRGETNOD, Error obtaining information for node [A]

Explanation: An error occurred whilst trying to look up node [A] in the DECnet database. The subsequent message gives more details.

%RTR-E-ERRINIACS, Unable to initialize tables

Explanation: Indicates that the configuration tables could not be initialized. The subsequent message gives more details.

%RTR-F-ERRJOUNAM, Error in journal file name - CREATE JOURNAL /SUPERSEDE and submit SPR

Explanation: An RTR journal file has been found which has an incorrect name format. This is probably an RTR bug, submit an SPR. This status may be returned by the CREATE FACILITY and SHOW JOURNAL commands.

Reissue the RTR CREATE JOURNAL command before restarting RTR.

%RTR-E-ERROPEFIL, Error opening file [A]

Explanation: Displays the name [A] of a file that the RTR utility was unable to open.

%RTR-F-ERROPEJOU, Error opening journal file, stv: [A]

Explanation: An error occurred while opening a journal file.

%RTR-E-ERRSTAACP, Unable to start ACP

Explanation: The RTR ACP process could not be started when a "START RTR" command was issued. The subsequent message gives more details.

%RTR-E-ERRSTARCH, Unable to start remote client handler

Explanation: The RTR remote client handler process could not be started when a "START REMOTE_CLIENT_HANDLER" command was issued. The subsequent message gives more details.

%RTR-E-EVTNAMILL, Unknown event name, [A]

Explanation: The "/EVENT" qualifier on the "open channel" command specified an unknown event name.

%RTR-E-EXPSYNILL, Expression has illegal syntax, [A] expected
n/[A]/

Explanation: The expression is invalid because token [A] was expected but [B] was found.

%RTR-E-EXPTOOCOM, Expression too complex
n[A]

Explanation: The expression [A] is too complex to evaluate. Please submit an SPR.

%RTR-S-FACADDED, Facility [A] added

Explanation: Displays the name of the facility that was successfully created after issuing an "CREATE FACILITY" command.

%RTR-E-FACALREXI, Facility already exists, [A]

Explanation: The facility [A] specified with the "CREATE FACILITY" command already exists.

%RTR-S-FACCREATED, Facility [A] created

Explanation: Displays the name [A] of the facility that was successfully created after issuing a "CREATE FACILITY" command.

%RTR-S-FACDELETE, Facility [A] deleted

Explanation: Displays the name of the facility that has been deleted after issuing a "DELETE FACILITY" command.

%RTR-S-FACEXTENDED, Facility [A] extended

Explanation: Displays the name [A] of the facility that was successfully extended (or created) after issuing an "EXTEND FACILITY" command.

%RTR-E-FACNAMBLA, Facility name is blank

Explanation: An empty string was specified where a facility name was expected.

%RTR-E-FACNAMILL, Facility name [A] contains illegal character "[A]"

Explanation: Facility name [A] contains the illegal character [B]. Legal characters are capital letters ("A" to "Z"), numbers ("0" to "9"), dollar ("$") and underscore ("_").

%RTR-E-FACNAMLON, Facility name [A] is longer than 30 characters

Explanation: Facility name [A] is too long.

%RTR-E-FACNAMSTA, Facility name [A] does not start with a letter

Explanation: Facility name [A] does not start with a capital letter ("A" to "Z").

%RTR-E-FACTABFUL, The FAC table is full

Explanation: This message is displayed when an "CREATE FACILITY" command is issued. It indicates that the maximum number of FACILITY to LINK relations has been reached.

%RTR-S-FACTRIMMED, Facility [A] trimmed

Explanation: Displays the name [A] of the facility that was successfully trimmed after issuing a "TRIM FACILITY" command.

%RTR-E-FDBTABFUL, The FDB table is full

Explanation: This message is displayed when an "CREATE FACILITY" command is issued. It indicates that the maximum number of facilities has already been reached.

%RTR-E-FENAMELONG, Frontend name string length greater than RTR_MAX_FE_NAM_LEN

Explanation: Status return indicating that the supplied value for the frontend name string exceeded the permitted maximum of RTR_MAX_FE_NAM_LEN characters.

%RTR-E-FLDNFND, Field [A] not found

Explanation: Field not found in call to rtr_request_info.

%RTR-I-FORCEDEXI, Forcing RTR application exit, process [A], PID [A]

Explanation: Displays the name [A] and PID [B] of the processes that were forced to exit due to the execution of a "STOP RTR" command.

%RTR-F-FUNCNOTSUP, Function not supported

Explanation: Function not supported.

%RTR-I-GRPMODCHG, Group changed from [A] to [A]

Explanation: RTR group changed from [A] to [B]

%RTR-I-HIGBNDHEX, High bound is [A] (hex)

Explanation: Displays the high bound of the server key range.

%RTR-E-HLBNCPTBL, Format of help library [A] is unknown

Explanation: The specified help library has an unknown format.

%RTR-I-HLPNOTFND, No help available (help file [A] not found)

Explanation: The help file specified by the RTRHELP environment variable cannot be found or cannot be opened.

%RTR-E-ILLDEVTYP, Device [A] is unsuitable for journals

Explanation: RTR can only create its journal files on directory structured devices. Re-issue the "CREATE JOURNAL" command specifying a disk.

%RTR-E-ILLPARTCHAR, Legal characters are alphanumeric and under-score

Explanation: Illegal chars in partition name argument.

%RTR-E-ILLREMDEV, Device [A] contains a node specification

Explanation: RTR cannot create its journal files on remote systems. Re-issue the "CREATE JOURNAL" command for local disk.

%RTR-E-INSUFPRIV, Insufficient privileges to run RTR

Explanation: More privileges required to run the RTR utility.

%RTR-F-INSVIRMEM, Insufficient virtual memory

Explanation: The application was unable to allocate additional virtual memory.

%RTR-F-INVCHANAM, Invalid chanam argument

Explanation: Invalid chanam argument.

%RTR-F-INVCHANNEL, Invalid channel argument

Explanation: Invalid channel specified.

%RTR-F-INVDEVNAM, Invalid device name length

Explanation: Invalid device name length.

%RTR-F-INVDSDEF, Msglen not consistent with len derived from msgfmt

Explanation: Invalid DSDEF format argument.

%RTR-F-INVEVTNUM, Invalid evtnum argument

Explanation: Invalid evtnum argument.

%RTR-F-INVEVTRAN, Invalid evtnum range

Explanation: Invalid evtnum range.

%RTR-E-INVFILNAM, Invalid file name, [A]

Explanation: The specified filename [A] is invalid.

%RTR-F-INVFL4CLI, Invalid flag for client channel

Explanation: Invalid flag for client channel.

%RTR-F-INVFL4SRV, Invalid flag for server channel

Explanation: Invalid flag for server channel.

%RTR-F-INVFLAGS, Invalid flags argument

Explanation: Invalid flags argument.

%RTR-F-INVJOINTXID, Invalid join transaction argument

Explanation: If the RTR_F_OPE_FOREIGN_TM flag was specified in the call to rtr_open_channel, then the call to rtr_start_tx requires a join TX parameter. Also, the formatID field of the join TXID must be set to a valid value (not RTR_XID_FORMATID_NONE).

%RTR-F-INVKSLENGTH, Invalid ks_length argument

Explanation: Invalid ks_length argument.

%RTR-F-INVKSTYPE, Invalid ks_type argument

Explanation: Invalid ks_type argument.

%RTR-F-INVMSGFMT, Invalid format argument

Explanation: Invalid format argument.

Possible reasons include the use of an invalid character or expression in the format string, or a mismatch in the number of bytes specified by the format string and the message length argument.

%RTR-F-INVMSGLEN, Invalid msglen argument

Explanation: Invalid msglen argument.

%RTR-E-INVMSGSIZE, Signed and unsigned data must be 1, 2, or 4 bytes

Explanation:

%RTR-F-INVNUMSEG, Invalid numseg argument

Explanation: Invalid numseg argument.

%RTR-E-INVOBJCT, Specified object type invalid for managed object request

Explanation: Status return indicating that an invalid management object type was the target of an rtr_set_info( ) command. Check your program and the RTR Application Programmer's Reference Manual for valid managed object types.

%RTR-F-INVOP4CLI, Invalid operation for client channel

Explanation: Invalid operation for client channel.

%RTR-F-INVOP4SRV, Invalid operation for server channel

Explanation: Invalid operation for server channel.

%RTR-F-INVPKEYSEG, Invalid pkeyseg argument

Explanation: Invalid pkeyseg argument.

%RTR-F-INVRCPNAM, Invalid rcpnam argument

Explanation: Invalid rcpnam argument.

%RTR-F-INVREASON, Invalid reason argument

Explanation: Invalid reason argument.

%RTR-E-INVRMNAME, Invalid resource manager name

Explanation: Invalid resource manager name.

%RTR-E-INVSTATCHANGE, Invalid to change from current state to the specified state

Explanation: RTR SET TRAN command cannot change.

%RTR-F-INVWAKEUP, Invalid wakeup argument

Explanation: Invalid wakeup argument.

%RTR-E-ITMALREXI, There is already something displayed at x = [A], y = [A]

Explanation: Indicates that invalid coordinates were specified on a "DISPLAY" command within a display file. There already is an item at point [A], [B].

%RTR-F-IVKEYW, Unrecognized keyword - check validity and spelling
n [A]

Explanation: A keyword specified in a command is not valid for the command. The rejected portion of the command is displayed between backslashes.

%RTR-F-IVQUAL, Unrecognized qualifier - check validity, spelling, and placement
n [A]

Explanation: An invalid qualifier is specified.

%RTR-F-IVVERB, Unrecognized command verb - check validity and spelling
n [A]

Explanation: The first word in the command is not a valid CLI command or a name equated with a command. The rejected portion of the command is displayed between backslashes.

%RTR-W-JOUALREXI, Journal already created

Explanation: A previously existing journal was found.

This status may be returned by the CREATE JOURNAL command.

1) You have issued an RTR CREATE JOURNAL command without deleting the previous journal.

Corrective action:

a) Use the /SUPERSEDE qualifier with CREATE JOURNAL or b) Delete the old journal with the DELETE JOURNAL command.

%RTR-F-JOUFILMIS, RTR journal file missing - Find missing file or CREATE JOURNAL /SUPERSEDE

Explanation: One of previously initialized RTR journal files for a backend could not be found. This status may be returned by the CREATE FACILITY and SHOW JOURNAL commands.

Probable causes are:

(1) One of the disks being used for journalling is unavailable, or

(2) A user has inadvertently deleted an RTR journal file.

Corrective action, either:

(a) Bring the missing disk back on line, or

(b) Reissue the RTR CREATE JOURNAL /SUPERSEDE command to create a new journal (in this case any recovery information in the old journal is lost).

%RTR-F-JOUFORCHA, Journal format has been changed - CREATE JOURNAL /SUPERSEDE

Explanation: The journal file(s) found have an out-of-date format. This status may be returned by the CREATE FACILITY and SHOW JOURNAL commands after a new version of RTR has been installed on a system.

Corrective action: Issue an RTR CREATE JOURNAL /SUPERSEDE command.

%RTR-W-JOUINUSE, Journal is locked by another user

Explanation: The journal is currently in use by another user.

This status may be returned by the CREATE JOURNAL, and DELETE JOURNAL commands.

Corrective action, either:

Wait for the other user to complete, and reissue the command.

%RTR-E-JOUNOTAVA, Error during recovery ([A]) from [A] journal ([A])

Explanation: RTR transaction manager requested recovery from a remote journal, but the request could not be delivered to the node hosting the journal. In a non-clustered standby configuration, this indicates that local recovery after a server failure could not be completed. No user action required, since the transaction manager will attempt to send the recovery query again, once it has detected that the remote journal has become available.

%RTR-F-JOUNOTFOU, Journal not found

Explanation: No RTR journal files can be found.

This status may be returned by the CREATE FACILITY, DELETE JOURNAL and SHOW JOURNAL commands.

Either:

1) You have not issued an RTR CREATE JOURNAL command

2) All journal disks are offline

3) The journal files have already been deleted.

%RTR-F-JOUNOTINI, Journal has not been created - CREATE JOURNAL

Explanation: No RTR journal files can be found.

This status may be returned by the CREATE FACILITY and SHOW JOURNAL commands.

Either:

1) You have not issued an RTR CREATE JOURNAL command

2) All journal disks are offline

3) The journal files have been inadvertently deleted.

Corrective action, either:

a) Bring the journal disk on line

b) Issue an RTR CREATE JOURNAL command.

%RTR-S-JOURNALINI, Journal has been created on device [A]

Explanation: Confirms that the RTR journal has been successfully created on device [A] after issuing the "CREATE JOURNAL" command.

%RTR-S-JOURNALMOD, Journal has been modified on device [A]

Explanation: Confirms that the RTR journal has successfully modified the size requirements on device [A] after issuing the "MODIFY JOURNAL" command.

%RTR-E-KEYTYPEVAL, Key type value for [A] missing - try one of string, signed, unsigned

Explanation: The type clause of the key qualifier requires a value - try one of string, signed, unsigned.

%RTR-F-KNLEXECFAIL, Knl_exec( ) delivered pid of 0 (starting [A]) on node [A][A]

Explanation: Indicates that knl_exec( ) failed, when trying to start a command server.

%RTR-E-KRINUSE, Key range already in use - respecify the key

Explanation: A call to rtr_open_channel( ) to create a partition specified a key range that is already in use. Respecify the key range.

%RTR-E-KRINUSEINJNL, Journalled transactions preclude change of partition name

Explanation: Transactions exist in the journal that were processed whilst this key range was assigned a different name. This equates to an attempt to change the partition name, which is not allowed. Recover or delete these transactions first.

%RTR-I-LFILREOPE, Log_file[[A]] [A] reopened

Explanation: Log file [A] re-opened.

%RTR-E-LINKDISABLED, Link disabled - connection rejected

Explanation: The link to this node has been disabled at the remote partner. It can be enabled with the command 'set link/enable' at the remote node.

%RTR-I-LIVEAPPL, Process [A], PID [A] still attached to RTR

Explanation: Displays the name [A] and PID [B] of the processes that are still attached to RTR.

%RTR-I-LNKDSC, Link to node [A] disconnected

Explanation: This message is issued once per disconnected network link during the execution of the RTR DISCONNECT LINK command.

%RTR-E-LOADLIBFAIL, Unable to dynamically load the shared library

Explanation: The RTR ACP was unable to load the shared library dynamically.

%RTR-E-LOADSYMFAIL, Unable to find the symbol in the shared library

Explanation: The RTR ACP was unable to locate the symbol name in the lib.

%RTR-E-LOCKFAIL, Cannot obtain lock for resource [A]

Explanation: Failed to obtain lock. Enable logging for more information about error.

%RTR-S-LOGFILSET, Logging to [A]

Explanation: Displays which log files will be used after issuing a "SET LOG" command with the "/FILE" or "/OPERATOR" or both qualifiers.

%RTR-S-LOGFILSUP, Logging suppressed

Explanation: Indicates that logging is successfully suppressed after issuing a "SET LOG command" without a "/FILE" or "/OPERATOR" qualifier.

%RTR-E-LOGNOTACT, Logging not active

Explanation: Indicates that logging was not active when a "SHOW LOG" command was issued.

%RTR-F-MAXPARM, Too many parameters - reenter command with fewer parameters

Explanation: A command contained more than the maximum number of parameters allowed. This error can be caused by (1) Leaving blanks on a command line where a special character (for example, a comma or plus sign) is required (2) Using symbol names or logical names that, when substituted or translated, contain embedded blank characters (3) Failing to place quotation marks around a character string with embedded blanks.

%RTR-E-MAXTOOSMA, Maximum number of blocks may not be less than /BLOCKS

Explanation: The number of blocks specified with the /MAX_BLOCKS qualifier in the CREATE JOURNAL or MODIFY JOURNAL command was lower than the number of blocks specified with the /BLOCKS qualifier.

%RTR-E-MONCMPERR, Syntax error in command at line [A] in file [A]

Explanation: error found compiling a MONITOR definition file.

%RTR-E-MONNOTACT, Monitor is not active, first use the "MONITOR" command

Explanation: A SCROLL or PRINT command was issued before any monitor picture had been displayed.

%RTR-E-MSGDATILL, Unable to convert string [A] to [A]

Explanation: The string [A] could not be converted to data type [B]. The subsequent message gives the reason.

%RTR-E-MSGTOOBIG, The number of bytes in message exceeds the maximum of %u

Explanation:

%RTR-W-NAMERR, Node name to address lookup error

Explanation: Error encountered whilst looking up a node address - details follow in a subsequent message text.

%RTR-E-NDBTABFUL, The NDB table is full

Explanation: This message is displayed when an "CREATE FACILITY" command is issued. It indicates that the total number of different nodes specified with this and all previous "CREATE FACILITY" commands would exceed the limit specified with the "/LINKS" qualifier when RTR was started.

%RTR-E-NFW, Operation requires "SETPRV" privilege

Explanation: "SETPRV" privilege is required to execute a remote command.

%RTR-E-NOACTION, No object management action specified - check argument set_qualifiers

Explanation: Status return indicating that no action was supplied along with an object management request. Check your program.

%RTR-I-NOAPPSRV, No application server channels currently declared

Explanation: Indicates that no application server channels are currently declared, and hence the requested information cannot be output.

%RTR-E-NOBACKEND, No backends specified

Explanation: No backends were specified on a "CREATE FACILITY" command and the node where the command was executed was specified as being a router.

%RTR-S-NOCHANGES, No changes made

Explanation: The modifications specified resulted in no changes being required.

%RTR-F-NOCHANOPEN, No channel is currently open

Explanation:

%RTR-E-NOCLASS, If selitm specified, please specify a data-class

Explanation:

%RTR-E-NODECNET, Network unavailable

Explanation: DECnet was shutdown on local node when the RTR utility was being used.

%RTR-F-NODEFDEV, No default device found for journal creation

Explanation: The CREATE JOURNAL command failed because rtr was unable to find a suitable default device for a journal.

%RTR-E-NODISOLATED, Node isolated - connection rejected

Explanation: Autoisolation has isolated the node. Connection attempts from the isolated node are refused in this state. Connections can be enabled by issuing the following commands:

o on the isolating node(s): set link/enable <isolated-node> o on the isolated node: set node/noisolate

This status supersedes the V2 condition LINKSHUT.

%RTR-E-NODNA, DECnet specified for [A], but transport protocol unavailable or disabled

Explanation: DECnet was specified as required through use of a node name prefix ("dna." or a substitute), but no corresponding entry in the node database can be found. Add an entry for the indicated node to your local node name database or to you name server.

%RTR-W-NODNANAM, DECnet is installed, but no node name definition found for [A]

Explanation: DECnet is active and enabled on the local node, but the address lookup for the indicated node failed. Since other network transports are available, you may continue, but protocol failover to DECnet will not be possible following this condition.

%RTR-E-NODNOTFND, Node [A] not found in configuration - check spelling and case

Explanation: Unable to locate the referenced node in the current configuration. Might occur if the address of the node was changed since it joined the configuration and the name was not entered exactly as displayed in show link output.

%RTR-E-NODNOTKNO, Node not known, [A]

Explanation: The node [A] is unknown to DECnet.

%RTR-E-NODTXOPENSTRING, No rm open_string specified

Explanation: No "open_string" were specified on a "CREATE RM" command while trying to register an underlying RM (XA Resource Manager) with ACP.

%RTR-E-NODTXRMLIB, No dtx rm lib pathname specified

Explanation: No "xalib path" were specified on a "CREATE RM" command while trying to register an underlying RM (XA Resource Manager) with ACP.

%RTR-E-NOFACILIT, No facilities have been defined

Explanation: No facilities have yet been defined with the "CREATE FACILITY" command.

%RTR-E-NOFRONTEN, No frontends specified

Explanation: No frontends were specified on a "CREATE FACILITY" command and the node where the command was executed was specified as being a router.

%RTR-I-NOHLPENV, No help available (environment variable RTRHELP not found)

Explanation: The environment variable RTRHELP that defines where the RTR help files are is not defined.

%RTR-W-NOIPNAM, IP networking is installed, but no host definition found for [A]

Explanation: IP networking is active and enabled on the local node, but the address lookup for the indicated node failed. Since other network transports are available, you may continue, but protocol failover to TCP will not be possible following this condition.

%RTR-E-NOKEYSEGS, You must specify at least one keysegment - use /KEY1 - /KEY9

Explanation: Except for a callout partition, it is necessary to define the key range, so the absence of any key segment descriptors is an error.

%RTR-F-NOKEYW, Qualifier name is missing - append the name to the slash

Explanation: A slash character is on the command line but is not followed by a qualifier keyword name.

%RTR-E-NOLICENSE, No license installed

Explanation: Appropriate license is not installed for the required operation.

%RTR-F-NOLIST, List of parameter values not allowed - check use of comma (,)

Explanation: The command does not accept a parameter list.

%RTR-F-NONPRINTBLE, Command line contains a non-printable character

Explanation: Command line contains a non-printable character.

%RTR-E-NOOUTSTND, No outstanding operations on channel [A]

Explanation: A "SYS$SYNCH" command was issued but there were no outstanding operations on channel [A]

%RTR-F-NOPAREN, Value improperly delimited - supply parenthesis

Explanation: A value supplied as part of a parenthesized value list for a parameter, qualifier or keyword is missing a delimiting parenthesis.

%RTR-E-NORMXASWITCH, No dtx rm xa_switch name specified

Explanation: No "xaswitch" were specified on a "CREATE RM" command while trying to register an underlying RM (XA Resource Manager) with ACP.

%RTR-E-NOROUTERS, No routers specified

Explanation: No routers were specified on a "CREATE FACILITY" command and the node where the command was executed was specified as being a frontend or a backend or both.

%RTR-E-NORTRPROC, No processes using RTR

Explanation: No processes are currently using RTR.

%RTR-E-NOSUCHCHN, No channel matched [A]

Explanation: The requested channel [A] has not been declared.

%RTR-E-NOSUCHCOU, No counter matched [A]

Explanation: The requested counter [A] does not exist.

%RTR-E-NOSUCHDIS, No such monitor file, [A]

Explanation: The requested monitor file [A] does not exist.

%RTR-E-NOSUCHFACILITY, The specified or implied facility does not exist

Explanation: The requested facility [A] does not exist. See CREATE FACILITY.

%RTR-E-NOSUCHITM, Nothing displayed at x = [A], y = [A]

Explanation: Indicates that invalid coordinates were specified on a "CLEAR DISPLAY" or "SHOW DISPLAY" command. No item is displayed at point [A], [B].

%RTR-E-NOSUCHNOD, No such node, [A]

Explanation: The requested node [A] does not exist.

%RTR-E-NOSUCHPRC, No such process, process ID = [A] (0x[A])

Explanation: The process with PID = [A] does not exist or is not using RTR.

%RTR-E-NOSUCHPRCS, No such process, process ID = [A]

Explanation: The process with PID = [A] does not exist or is not using RTR. Like NOSUCHPRC, but used where the PID is only available as a string.

%RTR-E-NOSUCHPRT, No partition matched [A]

Explanation: The requested partition [A] does not exist.

%RTR-E-NOTCP, TCP specified for [A], but transport protocol unavailable or disabled

Explanation: TCP was specified as required through use of a hostname ("tcp." or a substitute), but no hostname entry can be found. Add an entry for the indicated node to your /etc/hosts file or to you name server.

%RTR-E-NOTHLIST, Nothing to LIST ...

Explanation: Nothing to list.

%RTR-E-NOTHTODIS, There is nothing to display

Explanation: Indicates that no items had been defined with the DISPLAY command when a "MONITOR" or "SCROLL" command was issued.

%RTR-W-NOTNEG, Qualifier or keyword not negatable - remove "NO" or omit n [A]

Explanation: The word "NO" preceded a qualifier or keyword, but the qualifier or keyword cannot be specified as a negative.

%RTR-E-NOTNESTEDTX, TX in progress is not nested

Explanation: A call to rtr_prepare_tx was called on a channel with an active TX that is not a nested TX. To start a nested TX, you must start the TX by calling rtr_start_tx with a valid jointxid parameter.

%RTR-E-NOTSAMTYP, All counters must have same type, [A]

Explanation: Counter [A] is not the same type as the other counters in the expression. All counters in an expression must be either 'process counters', 'facility counters', 'link counters' or 'node counters'.

%RTR-I-NOTSTACOMSRV, Could not start command server on node [A][A]

Explanation: Indicates that a command server couldn't be started on the specified node [A].

%RTR-F-NOVALU, Value not allowed - remove value specification

Explanation: A value has been specified for a qualifier that does not take a value. Remove the value specification.

%RTR-E-NUMCONILL, Numeric constant has illegal syntax, [A]

Explanation: The numeric constant [A] is invalid.

%RTR-W-OBSQUAL, Qualifier [A] is obsolete - value ignored

Explanation: An obsolete qualifier has been specified on a command line. The qualifier no longer has any effect, and the specified value will be ignored.

%RTR-E-ONLONENOD, Only one node allowed if process ID specified

Explanation: If a process ID is supplied on "MONITOR" command then only one node may be monitored.

%RTR-E-ONLYDISP, Only "DISPLAY" command allowed in this context
n[A]

Explanation: The specified display file contained a command [A] other than "DISPLAY". This is sometimes caused by forgetting the continuation character ("-") on the end of a line that is to be continued. Use MONITOR /VERIFY to find the incorrect command.

%RTR-W-OPENVMSQUAL, Qualifier [A] is not supported on this platform - value ignored

Explanation: A qualifier has been specified on a command line that is effective only on the OpenVMS operating system. The qualifier has no effect on the executing platform, and the specified value will be ignored.

%RTR-I-OPTSUPSED, The RTR V2.x option [A] is obsolete in V3.x

Explanation: The option is obsolete in this version of RTR.

%RTR-E-OWNNODMIS, Executing node [A] not specified as frontend, backend or router

Explanation: The node [A] where a "CREATE FACILITY" or "EXTEND FACILITY" command was executed was not specified as being a frontend, router or backend.

%RTR-F-PARMDEL, Invalid parameter delimiter - check use of special characters
n [A]

Explanation: A command contains an invalid character following the specification of a parameter, or an invalid character is present in a file specification.

%RTR-F-PARREQ, Missing parameter - supply required parameter
n [A]

Explanation: Missing parameter.

%RTR-E-PARTNAMELONG, Partition name too long

Explanation: Long partition name argument.

%RTR-E-PROCIDILL, Illegal process ID, [A]

Explanation: The specified process ID [A] has an invalid format. Process ID's are hexadecimal numbers with up to eight digits.

%RTR-E-PRTBADCMD, Partition command invalid or not implemented in this version of RTR

Explanation: Status return indicating that the ACP received a request for an unknown partition command.

%RTR-E-PRTBADFPOL, Unrecognised partition failover policy code

Explanation: Status indicating that an invalid value was specified for the partition failover policy.

%RTR-I-PRTCREATE, Partition created

Explanation: Message returned via rtr_open_channel( ) upon successful partition creation.

%RTR-E-PRTDEFNCONFLICT, Name and key information refer to different partitions

Explanation: A call to rtr_open_channel( ) specified both the partition name and key information, but these refer to different partitions.

%RTR-E-PRTDELCAN, Partition deleted - operation canceled

Explanation: Status used to terminate a pending operation when the partition is deleted prior to completion of the operation.

%RTR-E-PRTMODRMBR, Partition must be in remember mode on the active member

Explanation: Restriction: partition must be in remember mode on the active member before shadowing can be turned off. Also returned if an attempt is made to restart recovery whilst not in remember mode.

%RTR-E-PRTMODSUSP, Partition not suspended - please suspend and retry the operation

Explanation: Completion status indicating a partition command was rejected as the partition was not in the prerequisite state. Suspend the partition first.

%RTR-E-PRTNACTIVE, Partition cannot be deleted so long as it has active servers or transactions

Explanation: Partition cannot be deleted so long as it has active servers or transactions.

%RTR-E-PRTNAMINUSE, Partition name already in use - use another name

Explanation: A call to rtr_open_channel( ) specified a partition name that is already in use in another partition of the facility. Use another name.

%RTR-E-PRTNAMINUSEINJNL, Journalled transactions preclude change of partition key range

Explanation: Transactions exist in the journal that were processed whilst this name was assigned to a different key range. Recover or delete these transactions first.

%RTR-E-PRTNAMNOTFND, The named partition does not exist

Explanation: A call to rtr_open_channel( ) specified a partition name that does not exist

%RTR-S-PRTNCREATED, Partition created

Explanation: The requested partition was successfully created.

%RTR-S-PRTNDELETED, Partition deleted

Explanation: The requested partition was successfully deleted.

%RTR-S-PRTNEWFPOLS, Failover policy set

Explanation: Status indicating successful change to the partition failover policy.

%RTR-S-PRTNEWPOLWAIT, Failover policy stored - awaiting servers before taking effect

Explanation: Status indicating successful change to the partition failover policy, but the change will only take effect once servers have been started.

%RTR-S-PRTNEWPRIO, Backend priority set

Explanation: Status indicating successful change to the backend priority.

%RTR-S-PRTNEWPRIWAIT, Backend priority stored - awaiting servers before taking effect

Explanation: Status indicating successful change to the backend priority, but the change will only take effect once servers have been started.

%RTR-E-PRTNODNOTDEF, No definition found for a node in the list - see log file

Explanation: Status indicating absence of a definition for a node named in list of back ends for the partition.

%RTR-E-PRTNODNOTLST, Local node must be in the list of back end nodes

Explanation: Status indicating erroneous absence of the local node from the ordered list of back ends for the partition.

%RTR-E-PRTNOSRVRS, Partition has no servers - please start servers and retry

Explanation: Cannot perform the requested action until servers are attached to the partition. Start servers and retry.

%RTR-E-PRTNOTBACKEND, Partition commands must be entered on a backend node

Explanation: Restriction: partition commands must be entered on a BACKEND node.

%RTR-E-PRTNOTSUSP, Unable to resume partition that is not suspended

Explanation: Status indicating completion of partition resume command where the target partition was not suspended.

%RTR-S-PRTRESUME, Partition resumed

Explanation: Status indicating successful completion of partition resume command. Transaction presentation is now enabled.

%RTR-S-PRTRSRTRCVY, Partition recovery initiated

Explanation: Completion status indicating that partition recovery was manually initiated by the operator.

%RTR-E-PRTRUNDOWN, Partition is in rundown prior to deletion - no action taken

Explanation: Cannot perform the requested action since the partition is being deleted.

%RTR-I-PRTSHDOFF, Partition [A]:[A] shadow state set to off by operator [A]

Explanation: Written to log file in response to a user request to change the state of partition shadow state.

%RTR-I-PRTSUSCAN, Suspend operation cancelled - partition resumed by operator

Explanation: Status indicating that a pending partition suspend operation has been cancelled as a result of an operator command to resume the partition.

%RTR-S-PRTSUSPENDED, Partition suspended - use resume to restart transaction presentation

Explanation: Status indicating successful completion of partition suspend command. Transaction presentation is now halted. Use resume to restart.

%RTR-E-PRTSUSTMO, Suspend operation timeout - partition still suspending

Explanation: Status indicating that a time out condition was encountered whilst waiting for a partition to be suspended. The partition will still be suspending - resume the partition to restart presentation of transactions.

%RTR-E-PTRARDYSUSP, Partition already suspended

Explanation: Status indicating completion of partition suspend command where the target partition was already suspended.

%RTR-E-PTRSUSPENDING, Partition already suspending - awaiting completion of current transactions

Explanation: Status indicating completion of partition suspend command where the target partition was already suspending. Be more patient.

%RTR-E-RCHALRSTA, RTR remote client handler already started

Explanation: The remote client handler was already running when the "START REMOTE_CLIENT_HANDLER" command was executed.

%RTR-E-RCHNOTSTA, RTR remote client handler not started

Explanation: The remote client handler had not been started when a command was issued which requires it to be running.

%RTR-E-RCHWASSTO, RTR remote client handler has been stopped

Explanation: The remote client handler had been stopped when a command was issued which requires it to be running.

%RTR-S-REASONSTS, Reason status: [A]

Explanation: Displays the contents [A] of the RSNSTS field of the TXSB after calling a RTR V2 system service via the DCL interface.

%RTR-E-RMSTRINGLONG, Resource manager open or close string too long

Explanation: Resource manager open or close string too long.

%RTR-E-RTRALRSTA, RTR already started

Explanation: RTR was already running when the "START RTR" command was executed.

%RTR-S-RTRLOGENT, [A]

Explanation: The RTR LOG command was used to make an entry in the RTR LOG

%RTR-I-RTRNOTRUN, RTR not running

Explanation: Message created specifically for the STOP RTR command if RTR is not currently running, so that the IVP does not report a fatal message.

%RTR-E-RTRNOTSTA, RTR not running

Explanation: RTR had not been started when a command was issued which requires RTR to be running.

%RTR-S-RTRRCHSTART, RTR remote client handler started

Explanation: Indicates that remote client handler has been successfully started after issuing a "START REMOTE_CLIENT_HANDLER" command.

%RTR-S-RTRRCHSTOP, RTR remote client handler stopped

Explanation: Indicates that the remote client handler has been successfully stopped after issuing a "STOP REMOTE_CLIENT_HANDLER" command.

%RTR-S-RTRSTART, RTR started on node [A][A]

Explanation: Indicates that RTR has been successfully started after issuing a "START RTR" command.

%RTR-S-RTRSTOP, RTR stopped on node [A][A]

Explanation: Indicates that RTR has been successfully stopped after issuing a "STOP RTR" command.

%RTR-E-RTRWASSTO, RTR has been stopped

Explanation: RTR had been stopped when the command when a command was issued which requires RTR to be running.

%RTR-S-SETTRANDONE, %ld transaction(s) updated in partition [A] of facility [A]

Explanation: The requested set transaction command was successfully performed.

%RTR-W-SETTRANROUTER, Cannot process this command, coordinator router is still available

Explanation: RTR SET TRAN command cannot change because router is still available.

%RTR-F-SPUJOUFIL, Spurious RTR journal file found - remove extra file, or CREATE JOURNAL /SUPERSEDE and submit SPR

Explanation: A spurious RTR journal file has been found which does not correspond to the other journal files on the system. This status may be returned by the CREATE FACILITY and SHOW JOURNAL commands.

Probable cause:

System management error. A user has copied a journal file, or a disk containing a journal file. RTR can now see extra journal files, or copies, that do not belong in the set.

Corrective action, either:

(a) Check the log for the relevant filenames, and delete or move the spurious journal file or files, or

(b) Reissue the CREATE JOURNAL /SUPERSEDE command (in this case any recovery information in the old journal is lost).

%RTR-I-STACOMSRV, Starting command server on node [A][A]

Explanation: Indicates that a command server is being started on node [A].

%RTR-E-SUPCHAEND, Superfluous characters at end of expression, [A]

Explanation: The expression is invalid because the characters [A] at the end of the expression could not be interpreted.

%RTR-I-SYSSRVCOM, [A] completed on channel [A]

Explanation: Displays the name [A] of the operation that completed on a channel after issuing a "SYS$SYNCH" command.

%RTR-I-SYSSRVNOW, [A] posted with no wait on channel [A]

Explanation: Displays the name [A] of the operation that was issued with the "/NOWAIT" qualifier on channel [B].

%RTR-E-TEMPLATE_NOT_FE, Template link [A] valid for front-ends only

Explanation: Template link names are valid for front end roles only. It is an error to attempt to associate a template link name with the router or back end roles. A template link is link whose name contains one or more wild characters chosen from the set "*?%". '*' indicates a sequence of wild characters; '?' and '%' indicate an occurrence of a single wild character.

%RTR-F-TIOSYS_FAILURE, General failure in TIO

Explanation: General failure in terminal I/O.

%RTR-F-TIO_BADROWCOL, The terminal is defined as : rows = [A] and cols = [A]

Explanation: Cannot use the terminal; rows or columns is set to 0. For UNIX platforms, please use "stty -a" to check rows and columns. Then use a command like "stty rows 50 cols 132" to set them correctly.

%RTR-E-TOOBIG, [A] may not be greater than [A]

Explanation: The value of qualifier [A] must be less or equal to [B].

%RTR-F-TOOMANCHA, Too many channels already opened

Explanation: Too many channels already opened.

%RTR-E-TOOMANCHN, Too many channels

Explanation: Displayed when a "SYS$DCL_TX_PRC" command is issued and the channel table is full.

%RTR-F-TOOMANDIS, Too many disks specified in journal definition

Explanation: Explanation: Too many disks were specified in journal definition. The RTR journal can be defined to use up to a maximum of sixteen disks.

User Action: Issue the CREATE JOURNAL command specifying a smaller number of disks.

%RTR-F-TOOMANREC, Too many records for one entry in the journal

Explanation: An attempt was made to write more than 65534 records to one entry (transaction) in the journal.

%RTR-E-TOOMANYOBJ, Max DECnet objects exceeded, raise and retry command

Explanation: The executor limit on the number of DECnet connect objects has been exceeded. Please use NCP to raise the maximum number of objects on this node.

%RTR-E-TOOMANYSELITM, More then one selitm specified in data-class :[A]

Explanation: More then one selitm specified in data-class on call to rtr_request_info.

%RTR-E-TOOMUCDAT, Too much data to be monitored

Explanation: An attempt was made to monitor too many processes. Please submit an SPR.

%RTR-E-TOOSMALL, [A] may not be less than [A]

Explanation: The value of qualifier [A] must be greater or equal to [B].

%RTR-I-TRMCENTRYNFND, No termcap-entry for terminal-type : [A].

Explanation: No such entry was found in the termcap file for the terminal type. For UNIX platforms, please check that the environment variables "TERM" and/or "TERMCAP" (if used) are correct. Also check that the TERMCAP file (if used) has a valid entry for your terminal type. (NOTE: These entries are case-sensitive.) The default terminal type is vt100.

%RTR-E-TRNOTALL032, Not all routers are at the minimum required version of V3.2

Explanation: Cannot perform the requested action since not all routers are at a minimum version of V3.2

%RTR-F-TRUNCATED, Buffer too short for msg

Explanation: Buffer too short for message, message truncated.

%RTR-F-TXNOTACT, No tx currently active on chan

Explanation: No transaction currently active on channel.

%RTR-E-UICNOTGRP, UIC [A] cannot be used in GROUP mode

Explanation: A "SET MODE/GROUP" command was issued while running under system UIC (group one). System UIC's cannot be used in group mode.

%RTR-E-UNEXPEND, Expression ended before [A] encountered

Explanation: The expression is invalid because it terminated where when token [A] was expected.

%RTR-E-UNKNOWQUAL, Invalid qualifier keyword value - check your program

Explanation: Status return indicating that an unrecognised qualifier keyword value was supplied. Check your program, and refer to the RTR Application Programmer's Reference Manual for permissible values.

%RTR-F-UNRROUNAM, Unrecognised api routine name for CALL

Explanation: The parameter for the CALL command is the name (or part of a name) of an rtr api routine. This allows the user to type, e.g. rtr call accept instead of rtr call rtr_accept_tx. This message is issued if the user has specified a part of an api routine name that does not match the name of an rtr api routine.

%RTR-F-VALREQ, Missing qualifier or keyword value - supply all required values

Explanation: A value must be specified for the keyword or qualifier.

%RTR-F-VALTOOBIG, 0x[A] is too big for [A] byte number

Explanation: A value has been specified that cannot be stored in the number of bytes specified. Specify a smaller value or a larger number of bytes.

%RTR-E-VERMISMAT, RTR version mismatch

Explanation: Indicates that utilities and/or sharable images being used are intended for a different version of RTR to that which is currently running on the system.

This message can however be ignored when it is displayed after issuing the first "STOP RTR" command after having just installed a new RTR release.

%RTR-I-WFPROCESS, Waiting for [A] to start up

Explanation: Displays the name [A] of the operation that completed on a channel after issuing a "SYS$SYNCH" command.

%RTR-E-WILNOTALL, Wild cards not allowed

Explanation: Wildcards ("%" and "*") are not allowed.

%RTR-E-WTTR, Not in contact with sufficient router nodes - please retry later

Explanation: Returned by a set partition command when either inquorate or no routers available to process the command. Try again later when none of the above conditions exist.

E

RTR log messages

This appendix describes the various error messages that can be sent to the operator console or written to RTR's operator log file.

%RTR-E-ABODEAREQ, Transaction aborted that was started by client that has since exited

Explanation: Indicates that a transaction has been aborted that was started by a client that has since exited.

%RTR-E-ABODEASRV, Transaction aborted that was accepted by a server that has since exited

Explanation: Indicates that a transaction has been aborted that was sent to a server that has since exited.

%RTR-E-ACCERR, Rejecting connect attempt from unconfigured node [A]

Explanation: Node [A] which has not been configured as part of any RTR facility is trying to establish connection. This could be a misconfiguration problem, or simply a problem with the setting up of the DNS service if DECnet-OSI is running.

%RTR-F-ACPINSRES, The RTR ACP has insufficient resources: details in the rtr log file

Explanation: The RTR ACP was unable to perform an operation due to an unusual condition. This is most probably a resource issue, e.g. when the ACP cannot create a shared memory segment.

The RTR log file for contains more details of the problem.

%RTR-E-ALRDCNCTD, Remote node already connected

Explanation: This can be a reason for rejecting a connect request. Submit an SPR.

%RTR-E-ALRINPRGS, Connection already in progress

Explanation: This can happen if both ACPs simultaneously try to connect to each other Submit an SPR.

%RTR-E-BADENVVARIABLE, Environment variable [A] has bad value [A]

Explanation: Log file message indicating that an environment variable has been defined with an illegal value.

%RTR-E-BADIDSIZ, Bad node ID size [A] detected at 0x[A]

Explanation: This message indicates that errors have been detected in processing of an internal node identifier. The presence of this message indicates a serious problem in the configuration of the network name/address databases, and RTR will likely be unable to operate correctly. Quorum and fault tolerance will be adversely affected. Check all network databases for consistency of node & host name and address consistency.

%RTR-E-BADIDTYP, Empty node ID encountered at 0x[A]

Explanation: This message indicates that errors have been detected in processing of an internal node identifier. The presence of this message indicates a serious problem in the configuration of the network name/address databases, and RTR will likely be unable to operate correctly. Quorum and fault tolerance will be adversely affected. Check all network databases for consistency of node & host name and address consistency.

%RTR-E-BADNETMSG, Bad message received from [A] - check network hardware? Indications %u %u %u %u %u %u

Explanation: This message is logged when RTR in unable to interpret the content of a network message received from a remote RTR ACP process. Message corruption is the most likely culprit. If the condition persists, consider a network health check. The indication numbers printed at the end of the message have meaning to RTR support engineers only.

%RTR-I-BEINQUO, Backend is quorate in facility [A]

Explanation: The backend role now has quorum in the facility [A]

%RTR-W-BENOQUO, Backend has no quorum in facility [A]

Explanation: The backend role has lost quorum for facility [A]

%RTR-E-BEREPLAYQDELETED, Replay queue for BE deleted

Explanation: A transaction in progress on a backend has had its (replay) queue of replies to the client deleted. Please report to RTR Engineering.

%RTR-E-BMHDRVSN, Unrecognised broadcast from [A] for facility [A] - check network? Indications %u %u %u [A]

Explanation: A node has received an unrecognised broadcast event from the indicated node. If the sending and receiving nodes are running compatible versions of RTR, the cause of this might be message corruption. If the condition persists, consider performing a network health check.

%RTR-F-BRODISBLO, Broadcast message(s) discarded because of network blockage

Explanation: One or more broadcast messages had to be discarded because the network throughput is not fast enough. Reduce broadcast rate or increase communications link capacity.

%RTR-F-BRODISCAC, Broadcast message(s) discarded because of memory cache congestion

Explanation: One or more broadcast messages had to be discarded because local memory was exhausted. Reduce the rate at which broadcasts are sent, or increase the efficiency of broadcast processing by the recipient applications.

%RTR-F-BRODISLIN, Broadcast message(s) discarded because of link unavailability

Explanation: One or more broadcast messages had to be discarded because there is no logical link to the destination node.

%RTR-I-CLUENABLED, RTR cluster [A] is enabled using [A]

Explanation: Information message indicating whether RTR is making use of any specific cluster software.

%RTR-S-CNCTACCFR, Connection request from [A] accepted

Explanation: A connection request from RTRACP running on node [A] has been accepted.

%RTR-S-CNCTCFRM, Connection confirmed by [A]

Explanation: A connection request has been confirmed by RTRACP running on node [A]

%RTR-I-CNCTLOST, Connection to [A] lost

Explanation: DECnet connection with node [A] lost.

%RTR-W-CNCTREJBY, Connection request rejected by [A]

Explanation: RTRACP on node [A] rejected a connect request from this node.

%RTR-W-CNCTREJFR, Connection request from [A] rejected

Explanation: RTRACP on node [A] made a connect request, which had to be rejected.

%RTR-E-COMDEAREQ, Transaction committed that was started by client that has since exited

Explanation: Indicates that a transaction has been committed that was started by a client that has since exited.

%RTR-E-COMDEASRV, Transaction committed that was accepted by a server that has since exited

Explanation: Indicates that a transaction has been committed that was sent to a server that has since exited.

%RTR-I-COMJOUSEA, Commencing journal search for transactions on facility [A] needing recovery

Explanation: Journal search is starting. This message appears when RTR is started.

%RTR-E-COMSRVFAIL, Command server failed - diagnostics written to [A]

Explanation: An instance of the RTR command server process has failed - report occurrence together with supporting information on current command to RTR Engineering.

%RTR-I-CONNALIAS, Link [A] connected as [A]

Explanation: Support for internet tunnels allows for the configuration of links from which connections appear to originate with an source address other than that by which the local node is registered locally, for example, the connection may appear to originate from an pseudo-adapter address assigned by the tunnel server. The CONNALIAS message registers the acceptance of such a connection, and lists the names of the local and connecting IP addresses.

%RTR-S-CURRTR, Node [A] now a router for facility [A]

Explanation: A new current router [A] has been found for facility [B]

%RTR-E-CURRTRLOSS, Current router lost for facility [A]

Explanation: The router handling facility [A] for this frontend node has failed, or [A] has been deleted on the router. No user intervention is expected, an attempt is being made to reconnect to an alternate router, if one is available.

%RTR-I-DTXRECOV, Commencing DTX journal search for transactions needing recovery, log_id = %08X-%04X-%04X-%02X%02X-%02X%02X%02X%02X%02X%02X

Explanation: DTX Journal search is starting. This message appears when RTR is started.

%RTR-I-DTXRECOVDONE, DTX journal search completed, [A] transactions found

Explanation: DTX Journal search has completed. This message appears when RTR is started.

%RTR-I-DUMPOBJECT, [A]:
n[A]

Explanation: Object dumped to log.

%RTR-E-ENQDEAREQ, Server rtr_reply_to_client to a client that has since exited

Explanation: Indicates that a call to rtr_reply_to_client was made for a client that has since exited.

%RTR-I-FACEXTNBE, Facility [A] extended, with backend [A]

Explanation: The configuration of facility [A] has been extended to include node [B] as a frontend.

%RTR-I-FACEXTNFE, Facility [A] extended, with frontend [A]

Explanation: The configuration of facility [A] has been extended to include node [B] as a frontend.

%RTR-I-FACEXTNTR, Facility [A] extended, with router [A]

Explanation: The configuration of facility [A] has been extended to include node [B] as a router.

%RTR-I-FACLOSTBE, Facility [A] lost Backend node [A]

Explanation: A connection has been lost with backend node [B] on facility [A]

%RTR-I-FACLOSTFE, Facility [A] lost Frontend node [A]

Explanation: This node is no longer a current router on facility [A], for frontend node [B]

%RTR-I-FACLOSTTR, Facility [A] lost Router node [A]

Explanation: A connection has been lost with router node [B] on facility [A]

%RTR-E-FACNOTDEC, Facility name not matched

Explanation: Result of a connection attempt to a remote node specifying a facility that does not (yet) exist on the remote node. Check that the facility name and configuration matches on all nodes concerned. If connecting to a V2 system, facility names must be specified in upper case.

%RTR-I-FACSTART, Facility [A] started on node [A]

Explanation: Facility [A] initialized on node [B]

%RTR-I-FACSTARTBE, Facility [A] started on node [A] as Backend

Explanation: A connection has been established with backend node [B] on facility [A]

%RTR-I-FACSTARTFE, Facility [A] started on node [A] as Frontend

Explanation: This node is now a current router for frontend [B], facility [A]

%RTR-I-FACSTARTTR, Facility [A] started on node [A] as Router

Explanation: A connection has been established with router node [B] on facility [A]

%RTR-I-FACSTOP, Facility [A] stopped on node [A]

Explanation:

%RTR-I-FACSTOPPED, Facility [A] stopped on local node

Explanation:

%RTR-I-FACTRMBE, Facility [A] modified, [A] no longer a backend

Explanation: The configuration of facility [A] has been modified to exclude node [B] as a backend.

%RTR-I-FACTRMFE, Facility [A] modified, [A] no longer a frontend

Explanation: The configuration of facility [A] has been modified to exclude node [B] as a frontend.

%RTR-I-FACTRMTR, Facility [A] modified, [A] no longer a router

Explanation: The configuration of facility [A] has been modified to exclude node [B] as a router.

%RTR-I-IGNREJACCEPTTX, Ignoring recovered aborted part TX for committed TX

Explanation: Status indicating that a part TX in aborted state was recovered from journal. However, the TX is already in committed state on the recovering node, so the aborted part-TX is ignored. This can happen under unusual circumstances such as where the TX has been rejected on one BE and on another BE the same TX has been later accepted (whether because of application inconsistency, or due to some condition in RTR such as journal

full). RTR has ensured that the TX has been committed, but the operator should nevertheless check the condition on the BE where the TX was aborted to determine why this occurred (possible resource problems on the server, for example).

%RTR-F-INCOMPAT, Incompatible RTR versions

Explanation: Attempt to start up an incompatible version of RTR on the same network with shared facilities.

%RTR-F-INTERFERENCE, Group/system interference. Start RTR from other account

Explanation: Internal error. Send SPR

%RTR-I-JOUEXCWRIT, Exception written to journal for transaction [A]; previous state = [A], reason status = [A]

Explanation: Indicates that an exception record has been written for the specified transaction. Manual intervention is required in order to see the transaction through to completion.

%RTR-W-JOUFILFUL, RTR journal file full - use MODIFY JOURNAL to increase size

Explanation: The RTR journal file is becoming full. Either reduce the number and size of concurrently active transactions, or increase the size of the journal file using MODIFY JOURNAL /MAXIMUM_BLOCKS

%RTR-F-JOUHDRERR, RTR journal record header error - CREATE JOURNAL /SUPERSEDE and submit SPR

Explanation: An inconsistency has been found in a record header within the RTR journal.

Corrective action:

1) Reissue the RTR CREATE JOURNAL /SUPERSEDE command.

2) Restart RTR.

3) Submit an SPR.

%RTR-E-JOUNOTAVA, Error during recovery ([A]) from [A] journal ([A])

Explanation: RTR transaction manager requested recovery from a remote journal, but the request could not be delivered to the node hosting the journal. In a non-clustered standby configuration, this indicates that local recovery after a server failure could not be completed. No user action required, since the transaction manager will attempt to send the recovery query again, once it has detected that the remote journal has become available.

%RTR-F-JOUOVERFL, RTR journal file overflow - transaction recovery information lost

Explanation: The journal file is so full that some transaction recovery information has had to be discarded. One or more of the currently active transactions may be incorrectly recovered if a system failure occurs in the near future.

%RTR-I-JOUSEACOM, Journal search on facility [A] completed. [A] recoverable transactions found

Explanation: Journal search has completed. This message appears when RTR is started The number of transactions needing recovery is indicated by [A].

%RTR-F-JOUSEQERR, RTR journal record sequence error - CREATE JOURNAL /SUPERSEDE and submit SPR

Explanation: An inconsistency has been found in the record sequence within the RTR journal.

Corrective action:

1) Reissue the RTR CREATE JOURNAL /SUPERSEDE command.

2) Restart RTR.

3) Submit an SPR.

%RTR-E-LINKSHUT, No longer accepting connect requests from this node

Explanation: A connection could fail between a router and a frontend or a backend if the link is in the "closed" state. This could happen if the link has been suspected of causing congestion to the rest of the network. This state can also be reset by system manager intervention.

%RTR-F-LNQOVERFLOW, LNQ table has overflowed

Explanation: This status is used to indicate an inadequacy in the static reservations of the per-link congestion queue header blocks. Send an SPR.

%RTR-E-LOADLIBFAIL, Unable to dynamically load the shared library

Explanation: The RTR ACP was unable to load the shared library dynamically.

%RTR-E-LOADSYMFAIL, Unable to find the symbol in the shared library

Explanation: The RTR ACP was unable to locate the symbol name in the lib.

%RTR-I-LOGFILENT, [A] [A] [A]

Explanation: Displays the time/date [A], node name [B] and user name [C] associated with the subsequent error message.

%RTR-E-LRCERROR, Found a LRC error in incoming message

Explanation: The checksum, also known as Linear Redundancy Check, was wrong in an incoming message.

%RTR-W-MESFLOCON, Message flow congestion on link to node [A] for facility [A]

Explanation: RTR internode communication has become congested. New messages are waiting for the congestion to clear.

%RTR-F-NETSHUT, Network has been shutdown or has become unusable ([A]) - automatic retry will follow

Explanation: This status results when someone stops DECnet on a node running RTR Might also occur if the network fails or otherwise becomes unusable.

%RTR-E-NOCURRTR, Current router search failed for facility [A]

Explanation: None of the routers specified for facility [A] are currently connectable. The search will continue after a short interval.

%RTR-W-NODENOTCNFG, Node is not configured for the facility

Explanation: Result of a connection attempt to a remote node where the connecting link is not configured in the requested facility at the remote node.

%RTR-W-NODISOLAT, Isolating [A], node suspected of causing congestion

Explanation: The remote node [A] has been diagnosed as causing network congestion; RTR will isolate the node from the rest of the network.

%RTR-W-NOFECREDIT, No credit for FE connect acceptance

Explanation: The router has no credit left to accept any more frontends right now. Other routers can be tried.

%RTR-E-NOTCONFIGURED, RTR not configured on [A] to recognize this node

Explanation: This means that none of the facilities defined on the local node [A] defined a valid role for the remote node to permit a connection.

%RTR-E-NOTRECOGNISED, Node not recognized

Explanation: Result of a connection attempt to a remote node where no facility references the connecting link at the remote node.

%RTR-E-OBJNOTDECL, RTR network object could not be established - will try again later

Explanation: A log file entry indicating that RTR was unable to establish its network object, most likely because the network was not available. A subsequent entry gives more detail on the error. RTR will retry the operation later, but the operator should investigate the state of the network.

%RTR-E-OBJUNKNOWN, RTRACP not running on node [A]

Explanation: A connection failed because RTR had not been started or had died on the remote node [A].

%RTR-E-OVERFLOW, Table has overflowed, resize NCF

Explanation: This is a general status used to indicate an inadequacy in the static reservations for the tables. Send SPR, with RTRACP dump.

%RTR-I-PATHLOST, Node [A] unreachable, retrying

Explanation: Node [A] cannot be reached at present. This generally means that RTR is trying to reestablish a connection.

%RTR-E-PROTOCOL, Incorrect protocol in optional data

Explanation: Internal error in messages between RTRACPs. May be caused by network packet data loss or corruption, so consider a network health check. If condition persists, submit an SPR.

%RTR-E-PRTBADCMD, Partition command invalid or not implemented in this version of RTR

Explanation: Status return indicating that the ACP received a request for an unknown partition command.

%RTR-I-PRTCMDFRMBE, Command received for partition [A]:[A] from backend node [A]

Explanation: Log file message indicating the origin of a command.

%RTR-E-PRTDELCAN, Partition deleted - operation canceled

Explanation: Status used to terminate a pending operation when the partition is deleted prior to completion of the operation.

%RTR-W-PRTLCLRECEXIT, Partition [A]:[A] local recovery terminated by operator [A]

Explanation: Log message indicating that recovery wait override requested by operator

%RTR-I-PRTNEWFPOL, Failover policy for partition [A]:[A] set to [A] by operator '[A]'

Explanation: Log file entry indicating acceptance of an operator request to change the failover policy of the indicated partition. Also appears when RTR automatically switches to pre-V3.2 compatibility mode.

%RTR-I-PRTNEWFPOLBE, Failover policy change for partition [A]:[A] received from router [A], new policy '[A]'

Explanation: Log file entry indicating receipt by a backend of an operator request to change the failover policy of the indicated partition. Also appears when RTR automatically switches to pre-V3.2 compatibility mode.

%RTR-I-PRTNEWFPOLTR, Failover policy change for partition [A]:[A] received from backend [A], new policy '[A]'

Explanation: Log file entry indicating receipt by a router of an operator request to change the failover policy of the indicated partition. Also appears when RTR automatically switches to pre-V3.2 compatibility mode.

%RTR-I-PRTNEWPRI, Priority for partition [A]:[A] set to [A] by operator '[A]'

Explanation: Log file entry indicating acceptance of an operator request to change the backend priorities of the indicated partition.

%RTR-I-PRTNEWPRITR, Priority change for partition [A]:[A] received from backend [A], new priority [A]

Explanation: Log file entry indicating receipt by a router of an operator request to change the priority of the indicated partition.

%RTR-S-PRTRESUMED, Partition [A]:[A] resumed by operator [A]

Explanation: Log file entry indicating successful completion of partition resume command. Transaction presentation is now enabled.

%RTR-W-PRTRSTRCVY, Partition [A]:[A] recovery initiated by operator [A]

Explanation: Log message indicating that partition recovery was manually initiated by the operator.

%RTR-I-PRTSCANJNL, Partition %ld scanning journal for node id [A]

Explanation: RTR is accessing the journal for node [B] that may contain relevant recovery information for this key range.

%RTR-E-PRTSETFAILTR, Router unable to process command

Explanation: A partition set command failed at the router. An entry is written to the log file describing the problem. Message arguments are the facility name and the KR ID. A second message is written detailing the nature of the problem.

%RTR-W-PRTSHDRECEXIT, Partition [A]:[A] shadow recovery terminated by operator [A]

Explanation: Log message indicating that recovery wait override requested by operator.

%RTR-I-PRTSHDWOFF, Command is set partition shadow state off

Explanation: This message appears in the router log file indicating what sort of request has been received.

%RTR-I-PRTSHDWON, Command is set partition shadow state on

Explanation: This message appears in the router log file indicating what sort of request has been received.

%RTR-I-PRTSUSPCAN, Suspend cancelled for partition [A]:[A], operator [A] - partition resumed by operator

Explanation: Log file entry indicating that a pending partition suspend operation has been cancelled as a result of an operator command to resume the partition.

%RTR-S-PRTSUSPEND, Partition [A]:[A] suspended by operator [A]

Explanation: Log entry indicating that partition has become suspended as the result of operator intervention.

%RTR-E-PRTSUSPTMO, Suspend timeout for partition [A]:[A], operator [A] after [A] seconds - partition still suspending

Explanation: Log entry indication that a command to suspend a partition timed out. The partition will still be suspending - resume the partition to restart presentation of transactions.

%RTR-W-PRTWAITJNL, Partition %ld waiting for access to journal for node [A]

Explanation: RTR cannot access one or more journals that may contain relevant recovery information for this key range. The journal referenced could be one of many unavailable journals. Either RTR should be started up on the missing backend, or the facility should be trimmed and the server restarted.

%RTR-F-QAROVERFLOW, No more QARs left

Explanation: This status is used to indicate an inadequacy in the static reservations for the internal query acceptor records. Make a note of all QRM counters using SHOW RTR /COUNTER=QRM* Send SPR with RTRACP dump.

%RTR-F-QCROVERFLOW, QCR table has overflowed

Explanation: This status is used to indicate an inadequacy in the static reservations for the internal query context descriptors. Send an SPR.

%RTR-F-QIROVERFLOW, No more QIRs left

Explanation: This status is used to indicate an inadequacy in the static reservations for the internal query initiation descriptors. Make a note of all QRM counters using SHOW RTR /COUNTER=QRM* Send SPR with the corresponding RTRACP dump.

%RTR-F-RAEOVERFLOW, No more RAEs

Explanation: This status is used to indicate an inadequacy in the static reservations for the internal response acceptor elements. Make a note of all QRM counters using SHOW RTR /COUNTER=QRM* Send SPR with the corresponding RTRACP dump.

%RTR-F-RDEOVERFLOW, RDE table has overflowed

Explanation: This status is used to indicate an inadequacy in the static reservations for the internal response dispatch elements. Make a note of all QRM counters using SHOW RTR /COUNTER=QRM* Send SPR with the corresponding RTRACP dump.

%RTR-E-REQDIED, Client exited, incomplete transaction aborted

Explanation: Indicates that a client exited before completing a transaction.

%RTR-E-REQDIEDPREP, Client exited after calling rtr_prepare_tx, transaction unresolved

Explanation: Indicates that a client exited after preparing a transaction.

%RTR-E-REQDIEDVOT, Client exited after issuing rtr_accept_tx, awaiting transaction result

Explanation: Indicates that a client died after accepting, but before completion of a transaction.

%RTR-W-ROLESMISMATCH, Node role definitions do not match for this facility

Explanation: The facility exists, but the definition of the role for the remote node is not the same as the one in the local node. The system manager has probably incorrectly defined the facilities on the two nodes concerned.

%RTR-E-ROUTERUNAVAILTMO, FE discarded transaction due to router unavailability timeout

Explanation: In rare circumstances, a transaction can be aborted by an RTR frontend with the status RTR_STS_ROUTERUNAVAILTMO. This can occur if an RTR client application has accepted the transaction, but the link to the router is lost before the RTR frontend has received a response from the router. If no RTR router becomes available for more than 8 minutes, then the transaction will be aborted by the frontend. This is to prevent the transaction being duplicated by RTR, since the router will unanimously decide the outcome of accepted transactions that have lost their frontends after about 10 minutes. The correct procedure in this case is for the client application to check whether the transaction has been completed once the link to a router is re-established.

%RTR-I-RQEQUALS, Transaction client was [A] on node [A]

Explanation: Displays the process name of the client [A] and its node address [B] in messages relating to transactions.

%RTR-W-RSPFAC, Response from Node [A] about Facility [A]

Explanation: A negotiation with remote node [A] about facility [B] has failed for the reason reported in the following line. This may be a reason for system manager intervention.

%RTR-W-RSPNODE, Connection to node [A] failed : reason is

Explanation: A negotiation with remote node [A] about facility [B] has failed for the reason reported in the following line. This may be a reason for system manager intervention.

%RTR-F-RTRACPFAIL, RTR ACP failed - diagnostics written to [A]

Explanation: The RTR ACP process has failed - report occurrence together with supporting information on current command to RTR Engineering.

%RTR-I-SETTRAN, RTR SET TRAN command entered facility:[A] partition:[A] tid:[A] state:[A] new_state:[A] since:[A] before:[A] user:[A]

Explanation: Log file entry recording that an RTR SET TRAN command is issued.

%RTR-I-SETTRANEND, RTR SET TRAN complete with status:%ld and %ld transactions updated in partition [A] of facility [A]

Explanation: Log file entry recording that an RTR SET TRAN command is complete.

%RTR-E-SRVABOREC, Server aborted transaction recovery, check database consistency

Explanation: Indicates that a server aborted a transaction being recovered after an earlier failure.

%RTR-E-SRVDIEDCOM, Server exited after being told to commit, check commit was completed

Explanation: Indicates that a server exited after being told to commit a transaction.

%RTR-E-SRVDIEDREC, Server exited during transaction recovery, check database consistency

Explanation: Indicates that a server exited whilst performing recovery of transactions lost on an earlier failure.

%RTR-E-SRVDIEDVOT, Server exited after voting on transaction, awaiting transaction result

Explanation: Indicates that a server exited before completing a transaction.

%RTR-I-STATECHANGED, Transaction:[A] journal state is changed from [A] to [A]

Explanation: Log file entry recording transaction's tkj state has changed.

%RTR-I-TIMEEQUALS, Issued at [A]

Explanation: Displays the transaction start time [A] in messages relating to transactions.

%RTR-W-TOOMANYNETIDS, Too many net IDs for node '[A]' - check for and eliminate any unnecessary adapter/protocol combinations

Explanation: On a system configured to run multiple network protocols over multiple adpaters, RTR can run out of space to store and communicate the resultant node IDs. You may be able to operate under this condition, but we recommend you review the system configuration and eliminate any unnecessary adapter/protocol combinations.

%RTR-I-TRINQUO, Router is quorate in facility [A]

Explanation: The router role now has quorum in the facility [A]

%RTR-W-TRNOQUO, Router has no quorum in facility [A]

Explanation: The router role has lost quorum for facility [A]

%RTR-I-TXIDEQUALS, Transaction ID = [A]

Explanation: Displays the transaction id [A] in messages relating to transactions.

Index

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X